IN THE UNITED STATES DISTRICT COURT
FOR THE SOUTHERN DISTRICT OF NEW YORK

INTERNATIONAL BUSINESS MACHINES CORPORATION, **Plaintiff,** **-vs.-** **PLATFORM SOLUTIONS, INC. and T3 TECHNOLOGIES, INC.,** **Defendants.**	**Civil Action No. 06 CV 13565 (LAK)**

PLAINTIFF IBM'S REPLY CLAIM CONSTRUCTION
BRIEF FOR THE *MARKMAN* HEARING

TABLE OF CONTENTS

Page

TABLE OF AUTHORITIES

Page

Cases

Statutes

Plaintiff International Business Machines Corporation ("IBM") respectfully submits this reply brief for the *Markman* hearing.[1]

I. INTRODUCTION

PSI's 158-page *Markman* brief is remarkably consistent in one respect − it seeks to limit the claims in IBM's patents to hardware implementations. For example, it seeks to limit "***processor***" to "***one or more integrated circuits***," and to limit "***floating point unit***" to "***a portion of a processor's circuitry that performs floating point calculations***." There is no mystery behind PSI's strategy. PSI's accused product is a software-based emulator, and it plans to use these constructions to argue that its software-based emulator cannot infringe claims that are limited to hardware implementations. That is undoubtedly why PSI's brief barely mentions its accused software emulator, even though the Federal Circuit is clear that the Court can usefully consider the accused product during claim construction.

Regardless of the motivation underlying PSI's unduly narrow constructions, the intrinsic and extrinsic evidence belie its strategy. IBM's claimed inventions are directed to improvements in computer architecture − whether implemented in hardware or software. Contrary to PSI's assertions, the inventions are not limited merely to hardware improvements, such as better circuits for mainframe computers. While the claims include references to system components, such as processors and registers and memory, they do so because these are functional components of computer architecture. This architecture is not limited to hardware. If the patents in suit were directed only to hardware, IBM could and would have drafted claim language directed to specific hardware components, such as circuits or transistors. *See, e.g., Linear Technology Corp. v. Impala Corp.,* 379 F.3d 1311 (Fed. Cir. 2004) (patent claims directed to "circuits" that regulate voltages in computers). IBM did not so narrow its claim language, and there is no intrinsic or extrinsic evidence that could support narrowing them now.

[1] For the Court's convenience, IBM has prepared an updated chart of the remaining patent claim terms/phrases in dispute, along with the parties' proposed constructions. The chart is attached as Exhibit 54 to the Supplemental Declaration of Edward DeFranco, filed herewith.

IBM engineers have invented better computer architectures that allow mainframes to run more efficiently, without sacrificing accuracy or reliability. As shown by the patents here, examples include multiple virtual computers created by emulation and partitioning a single physical computer, improvements to how floating point numbers are handled, more efficient sharing of memory resources, and a method for more efficiently restarting a program after a program interruption. IBM's inventions can be implemented in hardware or in a combination of hardware and software.

The '495 patent (the first patent addressed by the parties) demonstrates the problem of limiting these inventions to just hardware. The '495 patent is directed to "fully restoring a program context following an interrupt." The underlying invention ensures that the state of ongoing data processing (a program's "context") is saved when a program is interrupted, and then provides for execution of a novel instruction to restore that context when the program resumes. The claims refer to well-known computer system components ("processor," "register," and "instruction") because these are functional hardware and software components of computer architecture. But the invention is not merely an improvement to these individual components. The invention is a broader improvement to the overall computer architecture that includes – but is not limited to – these individual components. And it is certainly broader that what PSI puts forth with its hardware constructions.

Beyond its attempt to limit IBM's inventions to hardware, PSI also seeks to construe certain claim terms in a vacuum, without any support from intrinsic or extrinsic evidence, and apparently driven by PSI's planned non-infringement positions. For example, PSI's construction for the term "instruction" is limited to an instruction that "can be directly executed by ***the processor to which it is directed***." But there is no intrinsic or extrinsic evidence that could limit the construction of "instruction" in this way. PSI's construction is supported only by litigation strategy.

Repeatedly throughout its brief, PSI seeks to limit IBM's claims to the embodiments described in the patents. This is contrary to basic patent law. It is the claims that define the

scope of the invention, not the embodiments. The law encourages an inventor to claim an "invention" − not just the implementation of that invention in whatever equipment and tools the inventor may have used to reduce it to practice. The Federal Circuit is very clear on this point: "although the specification often describes very specific embodiments of the invention, we have repeatedly warned against confining the claims to those embodiments." *Phillips v. AWH Corp.*, 415 F.3d 1303, 1323 (Fed. Cir. 2005).

PSI also misapplies to five IBM patents the law for construing means-plus-function elements. PSI argues that these patents are invalid because there is "no structure" in the specification.[2] For this PSI relies heavily on *Aristocrat Tech. v. International Game Tech*., 521 F.3d 1328 (Fed. Cir. 2008), but misapplies this case for two reasons. First, PSI cannot present conclusory arguments on validity issues during claim construction – it should file a motion for summary judgment if it seeks to invalidate IBM's patents, which carry a statutory presumption of validity. Second, *Aristocrat* addressed the extreme case where a patent claiming software functions contained no algorithm, no flow chart, and no structure to support those software functions. Indeed, the patent in *Aristocrat* disclosed nothing more than "appropriate programming." *Id.* at 1334. In contrast, IBM's patents disclose both algorithms and flow charts that teach the structure necessary to implement the claimed functions. The role of claim construction is to identify the structure in the specification, not to render judgment as to whether it is legally sufficient or not. Finally, it defies common sense that the five different United States Patent and Trademark Office ("PTO") examiners who examined and allowed these patents would have overlooked the legal issue PSI now raises. PSI's extensive reliance on *Aristocrat* is misplaced.

[2] According to PSI, claims in the '495; '789; '678; '812; and '002 patents are invalid because they contain "no structure" to support claims written in means-plus-function format. (PSI Brief 40, 45, 60-61, 118, 135-36, and 148.) Before being allowed by the PTO, each patent was examined by a different examiner (and some by two examiners).

IBM has advanced constructions fully supported by the best intrinsic and extrinsic evidence. PSI's proposed constructions are not defensible, which perhaps explains why the majority of PSI's Brief is devoted to unsuccessful attempts to attack IBM's constructions rather than to convincingly explain why PSI's proposals should be accepted. The Court should adopt IBM's proposed constructions, which best reflect the inventions conceived by IBM's engineers.

II. FOUR DISPUTED CLAIM TERMS COMMON TO SEVERAL PATENTS

A. "Processor"

PSI agrees with IBM that a processor interprets and executes instructions. (*See* PSI Brief 8-9.) As a result, the parties' only dispute is whether a processor should be limited to "one or more integrated circuits." PSI plainly intends to use this construction to assert that its emulator does not infringe because it does not, under PSI's view, contain just "integrated circuits." That is improper under the evidence and the law, and the Court should adopt IBM's construction.[3]

PSI incorrectly contends that IBM's construction excludes the presence of integrated circuits. To the contrary, IBM's position is that a processor ***can*** contain integrated circuits ***as well as*** many additional components, including microcode, that are used for interpreting and executing instructions. It would be improper to have a construction for "processor" that only specifically sets forth one hardware element − an integrated circuit. That construction is misleadingly incomplete, and highlights why IBM's construction is correct.

For example, there is no dispute that microcode is part of a processor. (PSI Brief 8.) Rather than generally include microcode in its construction, PSI takes great pains to equate microcode with hardware, that is, code that is stored "within the larger integrated circuits that make up the processor." (PSI Brief 15.) But it is simply incorrect to equate microcode with

[3] PSI criticizes the testimony of the '520 inventor relating to his understanding of the term processor. However, PSI did not ask the inventor about his understanding of IBM's construction for processor, but rather whether other system components, such as keyboards and monitors, would fall within the broad construction for processor he offered ("anything that processes instructions"). (Mallick Depo. at 53:24-25.) PSI knows full well that keyboards and monitors do not fall within IBM's construction for processor.

hardware, as demonstrated by the intrinsic and extrinsic evidence. The '261 patent, for example, explicitly states that microcode "is software which is protected from being changed by user software."[4] ('261 patent, col. 8:63-65.)

In asserting that microcode is built into the integrated circuits and is a "manufactured component of the system," PSI relies on a portion of the '495 specification which states that microcode "can be considered to be a manufactured component of the system." ('495 patent, col. 4:57-58.) That is not the same as saying that microcode is hardware.

PSI cites another large passage from the '495 specification, including the quote above, to support its position that microcode is hardware. (PSI Brief 14; *see also*, '495 patent, col. 4:23-67.) A close reading of that passage makes plain that it distinguishes microcode from hardware throughout (*e.g.*, "microcode and hardware agencies"; "microcode and hardware of the system"; "Below the microcode level of control, the hardware elements of the system must have complete access to all system facilities ..."; "performed at either the microcode or hardware level"). The passage also distinguishes microcode from user software. Microcode provides an architectural interface to interpret and execute instructions, while user software is composed of the instructions that are interpreted and executed by the architectural interface. PSI tries to blur the differences between microcode and user software and asserts that IBM's construction covers both to show that it is overbroad. (PSI Brief 14, 17-18.) But IBM does not (and never did) contend that its construction includes user software.

Microcode is in fact a key component in a processor responsible for interpreting and executing instructions. For example, the '495 specification states that "[a]s is conventional in the art, CPU 102 has an instruction decoder for ***decoding instructions*** being executed as well as an execution unit for ***executing*** the decoded ***instructions***. These may be ***implemented by any suitable combination of hardware and microcode*** in a manner well know in the art." ('495

[4] PSI quotes this very language and then inexplicably argues it shows that microcode is not software. However the '261 patent could not be more clear that microcode is, in fact, software.

patent, col. 7:23-28, emphasis added.) There can be only one reading of the intrinsic evidence: microcode and hardware are distinct elements, and a processor can be composed of both. PSI's construction of "processor" fails to properly account for microcode and other software that could be used to implement an architectural interface. This is inconsistent with its concession that microcode is part of a processor.

On June 25, 2008, two days before IBM's Reply Brief due date and more than two months after PSI received IBM's Opening Brief, PSI proposed the following separate construction for processor: "One or more integrated circuits, including any microcode embedded in the circuits, that decodes and executes instructions." (Ex. 53.) However, this construction improperly limits "processor" to a narrow microcode implementation and ignores several other implementations well-known at the time of the inventions, examples of which are set forth below. Also, the terms "microcode" and "embedded" further complicate the construction, which is unhelpful to a jury.

PSI cites the IBM Principles of Operation stating that "[t]he physical implementation of the CPU may differ among models, but the logical function remains the same. The result of executing an instruction is the same for each model, provided that the program complies with the compatibility rules." (PSI Brief 11.) However the point of this statement is that, as one skilled in the art would know, a processor (or CPU) can be physically implemented in multiple different ways, such as with or without microcode, but its logical function of interpreting and executing instructions remains the same regardless of the physical model. (*See* Smotherman Decl. II ¶¶ 1-4.)[5]

The extrinsic evidence also clearly establishes that microcode is not hardware, and is not limited to being "part of the one or more integrated circuits that constitute the processor." (PSI

[5] IBM refers to the Declaration of Dr. Mark Smotherman In Support of IBM's Claim Construction Reply Brief herein as "Smotherman Decl. II." IBM also refers to the Declaration of Dr. Mark Smotherman In Support of IBM's Opening Claim Construction Brief as "Smotherman Decl. I."

Brief 15.) Microcode can be part of a processor integrated circuit, as is the case with certain models of the Intel x86 processor identified by PSI's expert.[6] (*See* Patt Decl., ¶ 9.) Microcode can also be stored in RAM, can be loaded from a disk, and can be updated/changed well after the processor is received by a customer. (Smotherman Decl. II, ¶¶ 5-12.) For example, the IBM ES/9000 Model 982 allowed for microcode updates and corrections, downloaded by means of a secure network to a system disk at any time. (Smotherman Decl. II, ¶ 12.) Dr. Smotherman's Declaration sets forth many other examples known in the art at the time of the inventions. (*Id.* at ¶¶ 5, 8-12.)

PSI's purported extrinsic evidence similarly shows that microcode is a kind of software (and not hardware). For example, PSI relies on the IBM Dictionary of Computing's definitions of "microcode" and "microinstruction" to support its position. However, all those definitions teach is that microcode is "implemented in a part of storage that is not program addressable," microcode is "***an alternative*** to hardwired circuitry," and microcode is composed of instructions at a lower level than machine instructions.

PSI also argues that the Court should not consider the first definition of "processor" from the IBM Dictionary of Computing because a separate standard body had not yet reached final agreement on the definition. However, such agreement is not required for the IBM Dictionary to constitute extrinsic evidence for what the term means. IBM's proposed construction uses the IBM Dictionary's first and thus more common and accepted definition. It is also the broader definition, and claim terms must be properly given their broadest reasonable meaning. *Rexnord Corp. v. Laitram Corp.*, 274 F.3d 1336, 1342 (Fed. Cir. 2001) ("In addition, unless compelled to do otherwise, a court will give a claim term the full range of its ordinary meaning as understood

[6] PSI states that IBM invited PSI's expert, Dr. Yale Patt, to serve as the Court's technical advisor. This is not correct. One of IBM's counsel, who is a personal friend of Dr. Patt's, contacted him to discuss technical advisor candidates but did not invite him to serve as a candidate, as PSI states.

by an artisan of ordinary skill."); *see also Free Motion Fitness, Inc. v. Cybex Intern, Inc.*, 423 F.3d 1343, 1348-49 (Fed. Cir. 2005) (citing *Phillips* for the same proposition).

The 1989 IBM-Intel License Agreement cited by PSI is irrelevant extrinsic evidence, as it is unrelated to the patents-in-suit. The Agreement expressly covers IBM patents "issued or issuing on patent applications entitled to an effective filing date prior to January 1, 1994." (PSI's Ex. 10 at 10.) All of IBM's asserted patents with the claim term "processor" have an effective filing date after January 1, 1994. In addition, the Agreement reflects a negotiated contract term for the purposes of the agreement at issue, not the meaning as understood by one of ordinary skill in the art at the time of the invention reading the claims and specifications.

B. "Instruction"

There are two disputed issues relating to the term "instruction": (1) whether an instruction is a "language construct" or a "string of digits"; and (2) whether an instruction must be "directly executed by the processor to which it is directed."

Contrary to PSI's assertion, IBM's use of the term "language construct" is supported by both the intrinsic and extrinsic evidence. (*See* IBM Opening Brief 17-19.) PSI itself recites a series of instructions from IBM's '520 patent which are language constructs (and not a string of digits):

> For example, a guest instruction 20 might be a memory-to-memory CISC instruction such as:
>
> ADD MEM1, MEM2, MEM3
>
> meaning "add the contents of memory location #1 to the contents of memory location #2 and store the result in memory location #3." A semantic routine 19 to emulate this guest CISC instruction might contain the following native RISC instructions:
>
> LOAD REG1, MEM1
>
> LOAD REG2, MEM2
>
> ADD REG3, REG2, REG1
>
> STORE REG3, MEM3

(*See* PSI Brief 69; *see also*, '520 patent, col. 4:45-63.)

PSI tries to distinguish a "language construct" from a "string of digits" and asserts that its proposal is proper because only a "string of digits" (and not a "language construct") can be executed by a processor. (PSI Brief 19-20.) But PSI cites no evidence that an "instruction" is limited to the precise form in which it is actually executed by a processor. In contrast, the intrinsic evidence shows the inventors considered "instruction" to be a broad term:

> Any of the six special emulation ***instructions*** may be ***implemented*** as ***microcoded instructions***. Other implementations may provide them as ***macro-instructions and have them replaced during compilation by in-line code*** to perform their functions in the appropriate target translation routines. Alternatively, they can be ***compiled as program calls*** to routines in the emulator provided to perform these functions when called.

('261 patent, col. 6:23-30, emphasis added.) Furthermore, the IBM Dictionary of Computing defines "macro-instructions" as "[a]n instruction ***in a source language*** that is to be replaced by a defined sequence of instructions in the same source language and that may also specify values for parameters in the replaced instructions." (Ex. 47, *IBM Dictionary of Computing*, 10th Ed., p. 409 (1994) (emphasis added).)[7] In addition, this Court has pointed out that: "whether directly or through the medium of another program, ***the sets of instructions written in programming languages*** – the source code – ultimately are translated into machine 'readable' strings of 1's and 0's, known in the computer world as object code, which typically are executable by the computer." *Universal City Studios, Inc. v. Reimerdes,* 111 F. Supp. 2d 294, 306 (S.D.N.Y. 2000) (emphasis added and footnotes removed).

Furthermore, IBM's construction is broad enough to include a "string of digits." For example, an instruction such as ADD A, B is instructing a processor to add the contents of registers A and B. Once the assembly code instruction (*i.e.*, ADD A, B) is compiled to machine code (*i.e.*, a string of digits), it remains a language construct (in the form of a string of digits) that instructs the processor to add the contents of registers A and B. (Smotherman Decl. II, ¶¶ 13-14; *see also* Smotherman Decl. I, ¶¶ 9-12.)

[7] "Ex. __" refers to the indicated Exhibit of the Declaration and Supplemental Declaration of Edward DeFranco.

To clarify this point and simplify this first disputed issue, IBM is willing to revise its construction to "a language construct, which can be a string of digits, that specifies an operation and identifies its operands, if any."

Regarding the second disputed issue, PSI does not provide any intrinsic or extrinsic evidence to support its limiting construction that an instruction "can be directly executed by the processor to which it is directed." PSI pulled this language out of thin air to establish a non-infringement argument – namely that in PSI's accused product an IBM instruction is never directly executed by the Itanium processor(s) to which it is allegedly directed. As explained in IBM's technology tutorial, PSI adds its emulation software to the Itanium processors to transform them into imitation IBM processors ("z/CPUs") so that they can execute IBM instructions. Again, as this Court has recognized, instructions may be executed directly or indirectly:

> Object code often is directly executable by the computer into which it is entered. It sometimes contains instructions, however, that are readable only by computers containing a particular processor, such as a Pentium processor, or a specific operating system such as Microsoft Windows. In such instances, a computer lacking the specific processor or operating system can execute the object code only if it has an emulator program that simulates the necessary processor or operating system or if the code first is run through a translator program that converts it into object code readable by that computer.

Universal City Studios, 111 F. Supp. 2d at 306.

PSI tries to sidestep the fact that its construction has no intrinsic support by arguing that certain instructions are synonymous with machine instructions, that those instructions are "executed," and that "[t]he parties agree that 'machine instructions' can be 'directly executed by a processor.'"[8] (PSI Brief 24.) However, none of PSI's cited intrinsic evidence refers to an instruction being "directly executed by the processor to which it is directed." PSI has added that phrase, and in particular the word "directly," to create another non-infringement argument. PSI

[8] IBM's construction for a machine instruction is a "string of digits that can be directly executed by *a* processor," not "a string of digits that can be directly executed by ***the processor to which it is directed***." As set forth in IBM's Opening Brief, the ability to be directly executed by *a* processor is quite different from the ability to be directly executed by ***the*** processor ***to which it is directed***. (IBM Brief 54-55.)

will contend that a person or ordinary skill in the art should understand an instruction to be "directly executed" by a processor when that processor executes the instruction using hard-wired circuits only, as opposed to with the help of microcode, semantic routines, emulation software, etc. Thus, if the Court accepts PSI's construction, PSI will then argue that an IBM instruction can never be "directly executed" by an Itanium processor, because it needs PSI's emulation software to execute the IBM instruction (just like other processors need microcode or millicode to execute certain instructions). But PSI's "directly executed by the processor to which it is directed" language appears nowhere in the intrinsic evidence, and the Court should reject it.[9]

C. "Program Status Word"

The parties' only dispute is whether a program status word must always be stored in a register. PSI admits that a "saved" program status word can be stored in something other than a register, but makes a convoluted argument that the "saved" program status word is "no longer the program status word," and only the "current" one is because only the current one "actually" directs the processor and indicates the next "actual" instruction to be executed. (PSI Brief 34.) The Court should reject this attempt to narrow the scope of the patent claims.

First, the disputed term in the '495 and '678[10] patent claims is "program status word" – not the "current" program status word, "saved" program status word, or some other type of program status word – and the construction must be broad enough to cover these different instances of program status words.

Next, PSI argues that the "current" program status word is the only program status word that actually directs the processor and indicates the next actual instruction to be executed. (PSI

[9] PSI also tries to justify its addition of this limiting language by construing the term "instruction" from the perspective of a particular processor. (PSI Brief 22.) In other words, PSI argues that an "instruction" is no longer an instruction if it is not understood by the particular processor to which it is directed. But PSI provides no evidence (or explanation) as to why "instruction" should be construed only from the perspective of a particular processor.

[10] The '678 patent claims only recite a "program status word" and not a "current" or "saved program status word."

Brief 34-35.) But this argument ignores the portion of the construction previously agreed to by the parties, which is that the program status word "includes the next instruction to be executed and includes the program condition code and program authority, where that data directs the processor in the execution ***of a program***." The program status word indicates the next instruction for a particular program, and directs the processor for that program. When a program status word is a saved program status word (and stored in a storage location or save area), it is still a program status word, still includes the necessary information about the next instruction to be executed for the interrupted program, and still includes the data to direct the processor in the execution of that interrupted program (*i.e.*, when the program is resumed).

PSI argues that a "'saved program status word' cannot be the program status word because there cannot be two different instructions that the processor is supposed to perform next." (PSI Brief 34.) But this ignores the entire premise behind the '495 patent, namely that one program can be interrupted by another program, thereby moving the program status word from the first program to a save area or storage location while using the program status word from the second program to indicate the next instruction. (*See* '495 patent, col. 1:7-12.)

PSI also asserts that the "intrinsic evidence ***unambiguously*** establishes that the program status word is a register" (PSI Brief 32), but this is simply false. PSI cites the IBM Principles of Operation, which says, "[t]he CPU provides registers ... [that] include the ***current*** program status word (PSW) ...." (PSI Brief 32, emphasis added.) However this quote supports IBM's construction, because it shows that only the current program status word is necessarily stored in a register. Further language from the same page is instructive:

> The program-status word (PSW) includes the instruction address, condition code, and other information used to control instruction sequencing and to determine the state of the CPU. The ***active or controlling PSW*** is called the ***current PSW***. It governs the program currently being executed.
>
> The CPU has an interruption capability, which permits the CPU to switch rapidly to another program in response to exceptional conditions and external stimuli. When an interruption occurs, the CPU places the ***current PSW*** in an assigned ***storage location***, called the ***old-PSW location***, for the particular class of interruption. The CPU fetches a ***new*** *PSW* from a ***second assigned storage location***. This ***new PSW*** determines the next program to be executed. When it has finished processing the interruption, the interrupting program may reload the

> ***old PSW***, making it again the ***current PSW***, so that the interrupted program can continue.
>
> There are six classes of ***interruption***: ***external, I/O, machine check, program, restart, and supervisor call. Each class has a distinct pair of old-PSW and new-PSW locations permanently assigned in real storage***.

(PSI Ex. 8 at 2-3, emphasis added) There are clearly multiple different types of program status words (current, old, new, saved) that are stored in multiple storage locations. This contradicts PSI's assertion that the program status word is limited to a register.

Accordingly, the Court should adopt IBM's proposed construction.

D. "Register"

The primary issue is whether the Court should adopt (1) IBM's proposed construction, which is consistent with the commonly understood meaning of the term (as presented, for example, in the contemporary IBM Dictionary of Computing) or (2) PSI's construction, which includes narrow limitations lacking any intrinsic support. IBM's proposed construction was created by IBM professionals and used by people of ordinary skill in the art at the relevant time , and should be adopted by the Court.

One of PSI's narrow limitations is that a register must be "in the processor." Contrary to PSI's arguments, registers ***can*** be inside processors, as described in the embodiment of the '495 and '789 patents, but they do not have to be. No intrinsic evidence supports the notion that registers can ***only*** be inside processors. The '495 patent, for example, is directed to "a method and apparatus for restoring the contents of the *CPU* registers." ('495 patent, col. 1:9-10, emphasis added.) Under PSI's construction, the modifier "CPU" before "register" would be redundant.

PSI, in an attempt to rebut IBM's construction, offers the analogy that "[y]ou could no more build a register out of 0s and 1s (*i.e.*, software) than you could build a bookshelf out of words." (PSI Brief 25.) PSI's analogy is misleading and unhelpful. "Internal storage," the term in the IBM Dictionary and IBM's proposed construction, includes hardware. It is also not an unbounded position. PSI claims that "this could be anything inside the computer, including the

hard drive," but this is at odds with the definition of "internal storage" from the IBM Dictionary because it excludes a hard drive ("storage that is accessible by a processor without the use of input/output channels").

PSI's construction additionally limits a register to something that "can be accessed faster than memory." PSI's arguments for this limitation are essentially a repeat of its arguments on "inside the processor." PSI dislikes the common meaning of the term in the IBM Dictionary, and so attempts to impose limitations that appear nowhere in the intrinsic evidence. It cites the '106 patent (which does not even use the term "register" in its claims) as specifically describing an embodiment of the '106 patent using "localized high-speed memory." This quote reveals the difficulty of PSI's proposed construction of "faster than memory." Because a register is memory, PSI's construction leads to the illogical position that memory is faster than memory. As IBM stated in its Opening Brief, because the intrinsic evidence on registers is silent on the subject of access speed, it is inappropriate to limit the construction of registers to a particular access speed. (*See* IBM Brief 24.) Notably, PSI failed to respond to this point.

Having exhausted the intrinsic evidence without finding any explicit narrowing or disclaiming beyond the ordinary meaning of the term, PSI turns to extrinsic evidence, particularly the inventor of the '520 patent. But this testimony is irrelevant, because the term "register" does not appear in the '520 claims. Moreover, the most PSI's evidence could show is that certain registers ***may*** be inside a processor, and this is entirely consistent with IBM's proposed construction.

The ***relevant*** extrinsic evidence supports IBM's position. U.S. Patents of the same time from Hewlett Packard, NEC, Intel, (all of which manufacture hardware that PSI has used) and IBM all refer to external registers that are outside processors:

- U.S. Patent 5,446,860 (Hewlett Packard): an apparatus comprising "a memory controller integrated circuit for controlling said memory modules, said external register being located external to said memory controller integrated circuit." (Ex. 48.)
- U.S. Patent 5,416,916 (NEC): a circuit containing: "an external register, external to said integrated circuit." (Ex. 49.)

- U.S. Patent 6,112,274 (Intel): "This particular embodiment comprises a processor 100 that may execute programs, including the interrupt handler program, and detect interrupt requests. The embodiment also comprises an interrupt storage location, that in this particular embodiment is an external register, the interrupt register 110." (Ex. 50.)

- U.S. Patent 6,223,196 (IBM): "a CISC microcontroller generally needs several clock cycles to write data to an external memory/register." (Ex. 51.)

Additional examples of architected registers in main memory would have been known by those of skill at the time of the invention. (*See* Smotherman Decl. II, ¶¶ 16-18.)

As a result, IBM's construction is more appropriate and the Court should adopt it.

III. THE INSTRUCTION/ARCHITECTURE PATENTS

A. The '495 "Resume Program" Patent

PSI's proposed constructions for the claim elements governed by 35 U.S.C. 112 ¶ 6 (and challenges to IBM's constructions) are based on several misapplications of Federal Circuit law. First, PSI's reliance on *Aristocrat Techs. Australia PTY Ltd. v. Int'l Game Tech.*, 521 F.3d 1328 (Fed. Cir. 2008) ("*Aristocrat/Int'l Game*"), is misplaced. In *Aristocrat/Int'l Game*, the parties disputed the construction for the claim element "control means." The parties agreed that this element was governed by 35 U.S.C. 112 ¶ 6, but disputed the corresponding structure. The patentee broadly proposed that the corresponding structure was "any standard microprocessor base [sic] gaming machine by means of appropriate programming" (*Id.* at 1333; citing the patent), while the accused infringer argued that this disclosure was insufficient and the patent therefore invalid. On summary judgment, the district court found the patent invalid.

The Federal Circuit affirmed that the patentee's disclosure of corresponding structure was insufficient. As the court stated, the patentee "contended that, in light of the breadth of the disclosure in the specification, any microprocessor, regardless of how it was programmed, would infringe claim 1 if it performed the claimed functions recited in the means-plus-function limitations of that claim." *Id.* at 1336. The court found that this "response reveals that Aristocrat is in essence arguing for pure functional claiming as long as the function is performed by a general purpose computer." *Id.* Indeed, the question in *Aristocrat/Int'l Game* was "not whether

the algorithm that was disclosed was described with sufficient specificity, but whether an algorithm was disclosed at all." *Id*. at 1337.

PSI relies on *Aristocrat/Int'l Game* to argue that "all of IBM's corresponding structure contentions should be rejected as improper functional claiming." (PSI Brief 37.) But PSI plainly ignores the format of IBM's disclosures of corresponding structure – including, but not limited to, IBM's use of "see, *e.g.*" – which are far more detailed than the disclosure of structure in *Aristocrat/Int'l Game*.[11] PSI argues that IBM is claiming ***any*** "hardware, software, or any suitable combination of the two" as its corresponding structure. This ignores IBM's specific references to structure in parentheses following that phrase as well as the algorithms disclosed that explain how to implement the recited function. For example, IBM's corresponding structure for the '495 patent's "means for decoding" element in claim 19 is "hardware, software, or any suitable combination of the two," ***as described by*** Fig. 1; 102; 5:41-11; or 7:21-28, and equivalents thereof for performing the claimed function ***with the algorithm described by*** Figs. 2A, 2B; 8:63-65, 9:5-15, and equivalents thereof.

With respect to IBM's use of "see, *e.g.*," it is black letter law that a means-plus-function element "shall be construed to cover the corresponding structure, material, or acts described in the specification ***and equivalents thereof***." 35 U.S.C. § 112 ¶ 6 (emphasis added). As such, IBM's use of "see, *e.g.*" simply refers to the fact that, in addition to the corresponding structures/algorithms described in the specifications (*i.e.*, the parenthetical citations), the structures/algorithms corresponding to the claimed functions also include equivalents thereof.

IBM is not simply arguing, as did the patentee in *Aristocrat/Int'l Game*, that "no disclosure of specific algorithms was necessary" because one of skill in the art would be able to devise algorithms for performing the claimed functions. *Aristocrat/Int'l Game*, 521 F.3d at 1337.

[11] In its proposed means-plus-function claim constructions for corresponding structure, IBM generally adopted a format as follows: recited structure (support in specification) plus function (support and/or algorithm(s) in specification).

Rather, IBM specifically cites algorithms disclosed in the specifications for performing the recited functions. As such, PSI's application of *Aristocrat/Int'l Game* is simply incorrect.

Second, PSI's contention that the patents fail to disclose sufficient algorithms for performing the claimed functions misapplies the law. Specifically, PSI's notion about the sufficiency of the algorithm is quite different than the Federal Circuit's, which instead judges the disclosure from the perspective of those skilled in the art. Indeed, the Federal Circuit has considered this issue in three recent cases. First, in *Aristocrat/Int'l Game*, the court held that "the sufficiency of the disclosure of algorithmic structure must be judged in light of what one of ordinary skill in the art would understand the disclosure to impart." *Id*. at 1337. It went on to say that the patentee "was not required to produce a listing of source code or a highly detailed description of the algorithm to be used to achieve the claimed functions in order to satisfy 35 U.S.C. 112 ¶ 6." *Id.* Second, in *AllVoice Computing PLC v. Nuance Communications, Inc.*, 504 F.3d 1236 (Fed. Cir. 2007), the court held that "[i]n software cases, therefore, algorithms in the specification need only disclose adequate defining structure to render the bounds of the claim understandable to one of ordinary skill in the art." *Id.* at 1245. The third case held that 35 U.S.C. 112 ¶ 6 "does not require that a particular algorithm be identified if the selection of the algorithm or group of algorithms needed to perform the function in question would be readily apparent to a person of skill in the art." *Aristocrat Techs. Australia PTY Ltd. v. Multimedia Games, Inc.* 2008 WL 484449, at *4 (Fed. Cir. Feb. 22, 2008) ("*Aristocrat/Multimedia Games*").

Recently, a court modified its original claim construction order, which had rendered several claims invalid for failing to adequately disclose algorithmic structure, "in light of the Federal Circuit's recent decisions in" *Aristocrat/Int'l Game*, *AllVoice* and *Aristocrat/Multimedia Games*. *SPX Corp. v. Bartec USA, LLC*, 2008 WL 2117608 at *1 (E.D. Mich. May 20, 2008) (emphasis added). In its modified claim construction order, the court found that the patent at issue disclosed adequate algorithmic structure corresponding to the "means for generating modulated signals" even though "the specification does not say *how* the microprocessor goes about providing modulation." *Id*. at *9 (emphasis in original). Indeed, the court concluded that

"the reference in the specification of the '796 patent to a 'microprocessor [that] can provide the modulation to the frequency generator circuitry' adequately identifies an algorithm to one ordinarily skilled in the art, which distinguishes a general purpose microprocessor from a special purpose computer." *Id.* at *1. The court continued:

> The plain meaning of that term [algorithm] and prior case law suggested that the patentee had to disclose a fairly detailed series of steps (if not mathematical formulae and source code). However, *AllVoice,* [*Aristocrat*/]*Multimedia Games,* and [*Aristocrat*/]*International Game* together clarify that the algorithm requirement is not so burdensome; according to the Federal Circuit's analysis, the disclosure in the present case is sufficient.... The *All Voice,* [*Aristocrat*/]*Multimedia Games,* and [*Aristocrat*/]*International Game* decisions show that the extent of the algorithm requirement is determined by what the ordinarily skilled artisan would know.

Id. at *7 - *9.

The algorithms disclosed here clearly illustrate more than a general purpose processor to one of ordinary skill in the art. Thus, in contending that the patents do not adequately disclose algorithms because they "only ***describe*** the function rather than actually demonstrate how to carry it out" (PSI Brief 78, emphasis in original), PSI plainly ignores Federal Circuit precedent, which mandates that "the sufficiency of the disclosure of algorithmic structure must be judged in light of what one of ordinary skill in the art would understand the disclosure to impart." *Aristocrat/Int'l Game*, 521 F.3d at 1337.

Third, PSI's misapplies *B. Braun Med., Inc. v. Abbott Labs.*, 124 F.3d 1419 (Fed. Cir. 1997), and *Medical Instrumentation & Diagnostics Corp. v. Elektra AB*, 344 F.3d 1205 (Fed. Cir. 2003), when it argues that several elements lack structures that are "linked" to the claimed functions. The standard for determining corresponding structure is not nearly as high as PSI suggests. All that is required is that "one of skill in the art would understand the specification itself to disclose the structure." *Id.* at 1212.

In *B. Braun*, the claimed apparatus was a valve for medical IV devices. The function was for restraining a flexible disc so as to restrain sideways movement. *B. Braun*, 124 F.3d at 1224. The patent disclosed one structure clearly identified for performing this function. During litigation the patentee attempted to "shoehorn" in an unrelated structure for performing the

claimed function. This second structure was described in the patent but there was not, to one of ordinary skill, "***any*** indication that [the second structure] corresponds to the recited function." *Id.* at 1424-25 (emphasis in original).

Similarly, in *Medical Instrumentation* the court determined that a step found in a box described as a method of the invention, and not linked to the structure that was the subject of the claim, could not be considered corresponding structure. 344 F.3d at 1213. In fact, there was no evidence that the figure describing the method step "would be understood by one skilled in the art to refer to structure at all, let alone software for [the claimed function]." *Id.* at 1213.

The reason PSI argues that there is no corresponding structure is that the failure to set forth corresponding structure renders a 35 U.S.C. § 112 ¶ 6 claim invalid. *B. Braun*, 124 F.3d at 1424-25. Such a finding must be established by clear and convincing evidence:

> For a court to hold that a claim containing a means-plus-function limitation lacks a disclosure of structure in the patent specification that performs the claimed function, necessarily means that the court finds the claim in question indefinite, ***and thus invalid***. Because the claims of a patent are afforded a statutory presumption of validity, overcoming the presumption of validity requires that any facts supporting a holding of invalidity must be proved by ***clear and convincing evidence***.

Budde v. Harley-Davidson, Inc., 250 F.3d 1369, 1376 (Fed. Cir. 2001) (emphasis added). PSI has failed to meet this burden. As set forth below, PSI's conclusory statements that the structures disclosed in IBM's specifications are not "linked" to the claimed functions are not supported by any evidence. One of ordinary skill in the art would understand that the structures cited by IBM are meant to perform the claimed functions. As such, PSI has not met the clear and convincing evidence standard and its argument must fail.

1. "means for decoding a program instruction specifying a register selected from said set of registers, said register pointing to a save area containing a saved program status word and saved register contents" (claim 19)

First, to address PSI's concern that the phrase "hardware, software, or any suitable combination of the two" is unbounded structure, IBM is willing to substitute the word

"processor" for that phrase.[12] As discussed above, the '495 citations in parentheses following that phrase illustrate that a processor performs the functions recited in each element of this claim. The specification discloses the structure – the algorithm for the workings of the processor, and one skilled in the art would know the different ways a processor could be physically implemented. In addition, IBM has recited the algorithm (in parentheses following the recited function) for performing the recited function.

PSI relies on *Aristocrat/Int'l Game* to argue that IBM's corresponding structure is inadequate, but ignores the fact that IBM cites a processor, the structure for that processor, and the algorithm for performing the recited function. IBM's disclosure goes far beyond what was disclosed in *Aristocrat/Int'l Game* and provides a sufficient algorithm for one of ordinary skill in the art to implement the recited function.

PSI also argues that IBM's disclosure of corresponding structure does "not show any algorithm capable of performing the relevant function – namely, decoding the instruction." (PSI Brief 42.) But IBM has recited a processor, and cited specific language (‘495 patent, col. 7:21-28) that the processor "has an instruction decoder for decoding instructions." The remaining function (which is part of the decoding operation) is to specify the register that points to the save area with the saved program status word and the saved registers. To address this part of the recited function, IBM points to Fig. 2A, which illustrates the program instruction, and the description in the '495 specification of the specified register where that program instruction is located. Based on this language, a person of ordinary skill at the time of the invention would have understood how to decode the instruction and specify the register. *See Aristocrat/Int'l Game*, 521 F.3d at 1337 ("[T]he sufficiency of the disclosure of algorithmic structure must be judged in light of what one of ordinary skill in the art would understand the disclosure to impart.").

[12] IBM originally proffered the "hardware, software, or any suitable combination of the two" because of the parties' dispute regarding the term "processor."

PSI incorrectly asserts that "the step of decoding an instruction is a step that results in the processor obtaining the information shown in Figure 2A." (PSI Brief 43.) However, Fig. 2A actually illustrates the instruction itself, and the decoding step is to decode the instruction illustrated in Fig. 2A to identify the register pointing to the right save area.

Even if, as PSI asserts, Fig. 2B and col. 9:5-15 describe how the processor deals with the program instruction after it has been decoded, it still provides context sufficient for a person of ordinary skill to decode the instruction. *See Aristocrat/Int'l Game*, 521 F.3d at 1337.

2. "means response to said decoding means for executing said instruction, said executing means comprising" (claim 19)

First, PSI argues that "IBM does ***not*** identify the 'execution unit' as corresponding structure" (PSI Brief 46, emphasis in original), even though IBM specifically ***highlights*** and references the execution unit in its brief. (IBM Brief 28.)[13]

Second, PSI's contends there is no corresponding structure because "nothing in the specification indicates that the 'execution unit' mentioned in col. 7 is 'responsive' to any 'decoding means.'" (PSI Brief 46.) This is both false and irrelevant. First, it is black letter law that the claims are part of the specification, and that "the claims of a patent define the invention." *Phillips v. AWH Corp.*, 415 F.3d 1303, 1315 (Fed. Cir. 2005). This element of claim 19 specifically indicates that the "means for executing" is "respons[ive] to said decoding means." ('495 patent, col. 14:5-6.) Also, "[t]he law is clear that ***patent documents need not include subject matter that is known in the field of the invention*** and is in the prior art, for patents are written for persons experienced in the field of the invention." *S3, Inc. v. NVIDIA*, 259 F.3d 1364, 1370 (Fed. Cir. 2001) (emphasis added). Presumably PSI avoids discussing the proper test because this would require it to take the untenable position that one of ordinary skill would not know that instructions are executed after they are decoded.

[13] IBM inadvertently identified part of the corresponding structure as col. 7:21-24 in the "construction box," but correctly identified it as col. 7:21-28 in its argument directly following the "construction box."

Finally, PSI ignores the fact that this element itself "comprises" the final two means-plus-function elements (discussed in Sections 3 and 4 below), and consequently includes the proposed structure for those two elements, including the recited algorithms for executing the instruction. IBM set forth in its Opening Brief that the corresponding structure for those two elements readily applies to this element. (*See* IBM Brief 28.)

3. "means for accessing said save area using the contents of the register specified by said program instruction" (claim 19)

Once again, PSI analyzes IBM's supporting cites for "hardware, software, or any suitable combination of the two" (*i.e.*, a processor) in a vacuum, ignoring IBM's recitation of the algorithm set forth in the '495 specification for performing the recited function. The entirety of IBM's proposed construction must be considered, including the recitation of that algorithm.

PSI then argues that each part of the '495 patent's algorithm for "accessing the save area ... does not provide a corresponding structure for accessing the memory." (PSI Brief 49-52.) Just as it previously ignored the second part of IBM's construction when addressing the first part, PSI now ignores the first part where IBM cites the processor that *is* the structure for accessing the memory, in the manner defined by the '495 specification. For example, "Fig. 6 is a flow chart of the execution of a Resume Program (RP) instruction" ('495 patent, col. 6:57-58), and steps 601 (col. 10:42-45), 602 (col. 10:46-50), 604 (col. 10:54-59), and 606 (col. 10:62-66) explain how to implement the accessing part of that execution. Contrary to PSI's assertion, a person of ordinary skill at the time of the invention would have understood, based on this disclosure, how to implement the recited function within the bounds of the invention. *See AllVoice Computing*, 504 F.3d at 1245 ("In software cases, therefore, algorithms in the specification need only disclose adequate defining structure to render the bounds of the claim understandable to one of ordinary skill in the art.").

PSI also contends that "Figure 6 cannot be corresponding structure for 'accessing a save area'" because "[c]laim 19 establishes that the means for 'restoring' is ***separate and distinct from*** the means for 'accessing.'" (PSI Brief 51, emphasis in original.) However, PSI offers no case

law indicating that a single structure cannot perform more than one function. Indeed, the Federal Circuit has stated otherwise. *See Lampi Corp. v. American Power Products, Inc.*, 228 F.3d 1365, 1374 (Fed. Cir. 2000) (citing *Rodime PLC v. Seagate Technology, Inc.*, 174 F.3d 1294, 1305 (Fed. Cir. 1999) for the proposition that "a particular means may perform more than one function."). Furthermore, it makes sense that Fig. 6 is corresponding structure for both of these elements because, as described above, the "means for executing" element ***comprises*** both the "means for accessing" and "means for restoring."

4. "means for restoring said program status word and said register from the saved program status word and saved register contents contained in said save area to resume execution at the instruction address contained in said saved program status word with the program context defined by said saved program status word and saved register contents" (claim 19)

The parties agree that Fig. 6 is at least part of the corresponding structure for this "restoring" function. It is not clear if PSI agrees that steps 603, 605, and 607 of Fig. 6 specifically recite the "restoring" function, but PSI does agree to steps 603 and 605. (*See* PSI Brief 54.)

In addition to Fig. 6, IBM has cited as corresponding structure the written description accompanying steps 603, 605, and 607 (col. 10:50-53, col. 10:59-61, and col. 10:66-11:12). If PSI contends that ***only*** Fig. 6 is corresponding structure, and not the written description accompanying that figure, then that position would be at odds with *Texas Digital Systems, Inc. v. Telegenix, Inc.*, 308 F.3d 1193 (Fed. Cir. 2002), *overruled on other grounds by Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005), where the Federal Circuit rejected a similar argument:

> Likewise, the district court's identification of the corresponding structure was incomplete. The description in the specification of the structure corresponding to the recited function is not limited to Figures 3 and 4, as instructed, but also includes the written description accompanying these Figures.... Moreover, as Figure 3 and its accompanying text serve merely as overview for introducing and explaining Figure 4, the corresponding structures must necessarily be found in Figure 4.

Id. at 1212. Here, the passages IBM cites provide written description for the relevant steps of Figs. 6, and therefore serve as corresponding structure.

PSI also argues that the first part of IBM's construction ("hardware, software, or any suitable combination of the two (Fig. 1; 102; Col. 7:21-28; or 5:4-11)") is not corresponding structure because it purportedly covers "***any*** structure that performs the recited function– regardless of whether that structure is clearly linked to the function or even ***exists*** in the specification." (PSI Brief 54, emphasis in original) PSI ignores the fact that IBM has cited corresponding language from the '495 specification which sets forth the boundaries of this structure, along with the algorithm for performing the recited function. PSI also ignores the fact that this recitation of structure is not considered in a vacuum, but rather by one of ordinary skill in the art. *See Aristocrat/Int'l Game*, 521 F.3d at 1337. The '495 specification as a whole, along with the specific portions highlighted by IBM, provides the limitations necessary for a person of ordinary skill to understand the boundaries of the structure.

B. The '789 "Time Stamp" Patent

1. "usable as a current time of day clock value in real-time processing" (claims 1 and 33)

PSI asserts that IBM expressly abandoned claim scope to overcome prior art for this phrase during the prosecution of the '789 patent, and is estopped from asserting that scope here. (PSI Brief 56.) This assertion is based on a gross misrepresentation of the '789 prosecution history, and ignores passages that directly contradict PSI's positions. In fact, IBM's proposed construction is fully supported by the '789 prosecution history, as well as the other intrinsic and extrinsic evidence.

The parties' dispute centers on two elements: the meaning of "current time of day value" and the meaning of "real-time processing."

(a) "Current Time of Day Value"

PSI contends, based on IBM statements in its 2/10/2003 Response to Office Action, that "current time of day value" means that the value cannot be used at a later time. In the 9/26/2002 Office Action, the Examiner rejected the term "timestamp" under 35 U.S.C. § 112 ¶ 1 because according to the Examiner, the word "timestamp" was not sufficiently disclosed and defined in

the specification. (PSI Ex. 17, 9/26/02 Office Action, p. 2.) In response, the '789 inventors disputed the Examiner's rejection, but agreed to replace "timestamp" with equivalent language: "Although applicants believe that 'timestamp' is supported in the application, applicants have amended the claims to recite that the sequence value is ***usable as a current time of day value***, in order to further prosecution of this application." (PSI Ex. 18, 2/10/03 Preliminary Amendment, p 10, emphasis added.) To simplify the issue, IBM is willing to modify its construction by replacing "timestamp" with "clock value."[14]

In that same response, the '789 inventors defined what this new language means: "In one aspect, applicants' invention is directed to generating a sequence value that is unique across multiple operating system images and is ***usable as a current time of day value***. ***That is, the time of day value reflects the time at which time is requested***." *Id.* at p. 11 (emphasis added.)

This unambiguous statement to the PTO directly contradicts PSI's position that IBM restricted the scope of the claims to distinguish it from "the values [that] can be created at any time and then stored for later use." (PSI Brief 57.) Furthermore, the inventors went on to distinguish their invention from the Frey prior art reference, not because the values in Frey could be used later, but because the values were not themselves "usable as a time of day value." PSI's selective quotation from the '789 prosecution history is taken out of context and is therefore misleading. The entire quote reads:

> In Frey, information is taken from a timestamp in order to make the unique value; however, ***the unique value, once created, is no longer, in and of itself, a timestamp***. That is, the value generated in Frey, which has time information embedded therein, is not a value that is usable as a current time of day value. ***This is because in Frey there is no requirement that the created values represent the current time of day*** and that the values be usable as time of day values. The only requirement is that the values be unique.

(PSI Ex. 18, 2/10/03 Preliminary Amendment, p 10, emphasis added.)

[14] PSI argues that because the '789 inventors replaced the word "timestamp" with equivalent language, IBM has somehow sacrificed subject matter related to that word. However, PSI's argument is inconsistent with the '789 prosecution history and unsupported by any case law.

The '789 inventors' subsequent response to the PTO further contradicts PSI's position: "In one aspect, applicants' invention is directed to generating a sequence value that is unique across multiple operating system images and ***is used as a current time of day value*** by one or more processors of the computing environment. ***That is, the sequence value, as generated, reflects the time at which time is requested by a processor*** and the processor can use (*i.e.* act upon) that value in real-time." (Ex. 34, 3/3/04 Preliminary Amendment, pp 10-11, emphasis added.) This was the '789 inventor's final communication to the PTO before their patent was allowed.

This meaning is unambiguously preserved in the specification: "When the clock is in the set state, execution of the STORE CLOCK instruction or STORE CLOCK EXTENDED instruction causes condition code 0 to be set and ***the current value of the running clock to be stored***." ('789 patent, col. 10:1-4, emphasis added.)

The inventors consistently defined "current time of day value" to mean that the value reflects the time at which the time was requested – including in their final communication with the PTO that led to allowance. This precise language is included in IBM's proposed construction.

Finally, PSI's proposal is (again) inconsistent with the point of this invention – as PSI itself acknowledges. (*See* PSI Brief 55.)

(b) "Real-Time Processing"

PSI argues that IBM's use of the phrase "real-time processing" limits its construction so that the current time of day value can only be used at the time it was requested. To support this incredibly narrow construction, PSI offers misleading arguments about the meaning of "real-time processing" that are directly contradicted by clear language in the '789 prosecution history. Its first such argument refers to the 3/3/04 Preliminary Amendment in which IBM added the "real-time processing" limitation. In the Remarks section of that Amendment, the '789 inventors distinguished the Wanish prior art reference by pointing out that it would take time to extract the time value from Wanish's unique sequence value. (Ex. 34, 3/3/04 Preliminary Amendment, p 11.) This merely clarifies that the '789's sequence values must be themselves be time values, not composites that require an extra processing step to extract time values.

The '789 inventors also distinguished the claimed "real-time processing" from Wanish's "post-processing": "It is indicated in Wanish that the ID can be useful in analyzing operating system dumps (Wanish, p. 3820). However, ***this analysis is in post-processing, not in real-time processing***." *Id.* The extra step required in Wanish renders it unusable for processing during the run of an application.

PSI also quotes the IBM Dictionary for the definition of "real-time processing": "the manipulation of data that are required or generated by some process while the process is in operation." (PSI Brief 59.) However PSI completely ignores the word "generated" in that definition. The '789 inventors' statements that the "***sequence value, as generated, reflects the time at which time is requested***" is entirely consistent with the definition of real-time processing. (Ex. 34, 3/3/04 Preliminary Amendment, pp 10-11, emphasis added.) The '789 inventors also made clear that their sequence values are usable by a process while that process is in operation, because they do not require an extra step to extract the time value (as in Wanish). *Id.* This is also consistent with the IBM Dictionary definition. PSI's statement that "real-time" changes the meaning of "current" is a non sequitur. This definition nowhere addresses "current."

Finally, PSI invokes a "precision" requirement based on a discussion of "predictable resolution" in the '789 specification, which it then uses to support its baseless notion that the sequence value can only be used at the "precise" time it is requested. (PSI Brief 59-60.) But the intrinsic evidence makes plain that "precision" refers to the number of bits in the time of day clock (or extended time of day clock), where more bits (*e.g.*, 87 bits instead of 51 bits) represents a more precise time value. ('789 patent, col. 12:65-13:18.) This provides no support for PSI's alleged requirement that the sequence value can only be used at the precise time it is requested.

PSI's argument that IBM's construction should be barred by the '789 prosecution history is simply not supported by the evidence. IBM's construction is consistent with the '789 claims, specification, and prosecution history, and it should be adopted.

2. "at least one computer usable medium having computer readable program code means embodied therein for causing the generating of unique sequence values usable within a computing environment, the

computer readable program code means in said article of manufacture comprising" (claim 33)

As with all the other means-plus-function claim elements for which PSI alleges "no corresponding structure," here PSI has failed to meet its burden of establishing, by "clear and convincing evidence," that "the disclosure[s] ... do not constitute sufficient structure to define the claim terms for the ordinarily skilled artisan." *AllVoice Computing PLC,* 504 F.3d at 1245.

And as it has done for other means-plus-function claim elements, PSI also misrepresents IBM's recitation of corresponding structure for this claim. IBM is not proposing "***all*** 'articles of manufacture' containing any software that causes a general purpose computer to perform the function recited in the limitation." (PSI Brief 63.) Instead, IBM is proposing the article of manufacture set forth at col. 14:61-15:4 of the '789 specification and equivalents thereof, followed by the recited function and, in parentheses, the algorithm for performing the recited function through software on that computer program product.

The '789 specification describes the structure of the article of manufacture at col. 14:61-15:4. It is irrelevant whether this language appears in other IBM patents, because it plainly recites structure for the article of manufacture. A person of ordinary skill at the time of the invention would certainly understand the structure for a "computer useable media" or "program storage device readable by a machine" that stores at least one computer program of instructions executable by that machine (*i.e.*, software) for performing the recited functions.

PSI's argument that col. 14:61-15:4 "does not disclose ***what*** is recorded on that media" is misleading. (PSI Brief 64.) IBM's proposed corresponding structure includes this recitation of structure, as well as the algorithm for the software that performs the recited function. And this disclosure has nothing in common with the disclosure in *Aristocrat/Int'l Game*, where the only disclosure was "appropriate programming." 521 F.3d at 1333. PSI ignores the second part of IBM's construction when it addresses the first part, and vice-versa. Further, because this first element "comprises" the second and third elements, the structural analysis must take into account the structure set forth for those elements (addressed in Sections (a) and (b) below).

(a) "computer readable program means for causing a computer to provide as one part of a sequence value timing information comprising at least one of time-of-day information and date information" (claim 33)

PSI misreads IBM's recitation for how a person of ordinary skill would cause a computer to provide time and date information as part of the sequence value. As explained earlier, IBM's recitation does not consist of four examples or alternatives, but instead recites the corresponding disclosure in the '789 specification. For example, Fig. 10 of the '789 patent is a flowchart illustrating "one embodiment of the logic associated with constructing an extended time-of-day clock value." ('789 patent, col. 4:57-59.) The '789 specification explains Step 1002 of Fig. 10 ("SET BITS 8 TO S+8 TO VALUE OF RUNNING CLOCK"):

> In particular, bits 0-7 (i.e., the TEX) are set to zeros, STEP 1000. Further, bits 8 to s+8 are set to the value of the running physical clock, STEP 1002. In one example, the running clock is located in the CPU. In other embodiments, it is located elsewhere, such as within a shared memory bus.

('789 patent, col. 9:1-5.)

PSI complains that this passage (and IBM's other cited passages) "provide information only as to ***some*** of the tasks required for the function (*e.g.*, setting bits to the value of the running clock) but not others (*e.g.*, retrieving the value)." (PSI Brief 65-66.) But based on the cited passages, a person of ordinary skill at the time of the invention would have understood how to implement this function, including the alleged discrepancies proposed by PSI. *Aristocrat/Multimedia Games*, 2008 WL 484449, at *4 ("The law does not require that structure be explicitly identified as long as a person of ordinary skill in the art would understand what structure is identified in the specification."). PSI's argument is premised on its requirement that the specification disclose each and every detail to perform the recited function. *Aristocrat/Multimedia Games* and other Federal Circuit cases confirm that this is simply not the law.

Further, PSI's reliance on *Default Proof Credit Card Sys. v. Home Depot U.S.A., Inc.*, 412 F.3d 1291 (Fed. Cir. 2005) is inapposite. In *Default Proof*, the patentee argued that the following three structures within a point-of-sale ("POS") terminal amounted to corresponding

structure for the "means for dispensing at least one debit card" element: (1) a "kiosk" associated with the POS terminal; (2) the receipt printer peripheral portion of the POS terminal; and (3) the LCD or CRT display peripherals of the POS terminal. *Id*. at 1300-01. But, as the court pointed out, none of those structures "form part of the description of the POS terminal in the specification." *Id*. at 1301. Indeed, none of the terms proffered by the patentee - kiosk, printer, peripheral, LCD or CRT - "even appear in the [patent's] specification." *Id*. at 1301. The Federal Circuit agreed that even the POS terminal itself did not provide corresponding structure, noting that "[t]he intrinsic evidence demonstrates that the POS terminal and any structure for distributing debit cards exist separately. Both the structure and the language of claim 1 indicate that the point-of-sale assembly and dispensing means constitute separate components." *Id*. at 1299-1300.

Here, of course, IBM cites passages that both appear in the specification and are not separate components from the claimed function. As such, PSI's argument must fail.

(b) "computer readable program means for causing a computer to include as another part of said sequence value selected information usable in making said sequence value unique across a plurality of operating system images on one or more processors of said computing environment, wherein said sequence value is usable as a current time of day clock value in real-time processing by one or more processors of the computing environment" (claim 33)

The parties dispute whether the "wherein" clause should be construed as part of this means-plus-function element. PSI erroneously contends that IBM wants to "simply omit this unwanted limitation." (PSI Brief 63.) IBM relies on *Lockheed Martin Corp. v. Space Systems/Loral, Inc.*, 324 F.3d 1308 (Fed. Cir. 2003) for the proposition that the "wherein" clause should not be construed ***as part of the § 112, ¶ 6 function***. IBM has never proposed that it simply be "omitted" or is otherwise non-limiting. Indeed, the parties offer separate constructions for the phrase "usable as a current time of day clock value in real-time processing" and the term "processor." However PSI's position is unsound because it requires that the terms of the "wherein" clause be construed ***twice***.

Furthermore, the "wherein" clause recites that the entire sequence value is "usable as a current time of day clock value in real-time processing", while the means-plus-function element at issue is for "another part" of this sequence value. In fact, the means-plus-function element discussed above in part (a) covers one part of the sequence value, this means-plus-function element covers another part, and the wherein clause recites the result of using these two parts. *See Lockheed Martin*, 324 F.3d at 1319 (one of ordinary skill in the art would understand that this "whereby" clause "merely states the results of the limitations in [the] claim"). Thus, it makes no sense to include the "wherein" clause with this means-plus-function claim element.

PSI cites *Griffin v. Bertina*, 285 F.3d 1029 (Fed. Cir. 2002), for the proposition that "wherein" clauses and "whereby" clauses are somehow different when construed in the context of § 112 ¶ 6 limitations. *Griffin* was an interference case discussing limiting preambles and did not even consider § 112 ¶ 6. Instead, it addressed the issue of whether a "wherein" clause is limiting ***at all***. As discussed above, IBM is not arguing that the "wherein" clause is not limiting, but rather that it should not be construed as part of the § 112 ¶ 6 function for this element.

IV. THE EMULATION PATENTS

A. The '520 Address Translation Patent

1. "instruction set" (claims 1, 9)

PSI concedes that IBM's proposed construction for an "instruction set" is correct ("the complete set of operations of the instructions of a ***computer architecture***"), but asserts that an "instruction set" is also the complete set of operations of a ***computer***.[15] PSI's assertion is premised on a computer having one and only one instruction set, when in fact a computer can have more than one instruction set.

PSI argues the '520 intrinsic evidence shows that a "processor" and a "computer" are the same, and that it associates an instruction set with a processor (or CPU). A "processor" and a "computer," however, are not the same. A computer includes one or more processors as well as

several additional components (*e.g.*, memory, display device(s), I/O connections, etc.). A computer can also include processors designed for different instruction sets. For example, the Myriad computer system includes processors compatible with two distinct instruction sets. (Smotherman Decl. II, ¶ 15.) Other examples were readily apparent to one of ordinary skill in the art at the time of the invention. (*Id.*)

Indeed, PSI's accused product is yet another example of a computer that supports different instruction sets.[16] Contrary to what it asserts in its brief, PSI's own web site asserts that its mainframe computers run IBM z/OS and Itanium-based instruction sets (Ex. 52), and PSI's computers do in fact execute the IBM Test FP Data Class Instruction (described in the '678 patent).

PSI's contention that "IBM acknowledges that Intel-based computers cannot execute IBM instructions" is simply false. (PSI Brief 75.) As illustrated below, PSI crops two quotes from IBM's opening brief to misrepresent IBM's position (portion cited by PSI in bold):

- "Because the ***IBM instructions are not compatible with the non-IBM architecture of the Intel-based computers***, PSI claims its layer of software translates each IBM instruction so that it can be understood and executed by the Intel computer." (IBM Brief 6-7.)

- "Overall, PSI's goal is to enable computer programs based on IBM's z/Architecture and ESA/390 Architecture to run on computer systems that are based on very different architectures – Intel's Itanium Architecture – and to allow those ***Intel-based computers*** (which otherwise ***are incapable of understanding and executing IBM instructions***) to mimic and appear to function as IBM mainframes." (IBM Brief 7.)

In fact, PSI itself claims that its computers run IBM's z/OS operating system. (Ex. 52.)

PSI's proposed construction is also inconsistent with positions this Court has taken:

> Object code often is directly executable by the computer into which it is entered. It sometimes contains instructions, however, that are readable only by computers

[15] This usage is inconsistent with PSI's own expert, who in defining "Instruction Set Architecture" never mentions the term "computer" at all. (Patt. Decl., p. 12.)

[16] On June 25, 2008, PSI offered to accept IBM's construction if IBM agreed "not to argue that a computer system can have two or more instruction set architectures." (*See* Ex. 53.) However, IBM cannot agree to this proposal because PSI alleges that its ***own*** system supports multiple instruction set architectures.

> containing a particular processor, such as a Pentium processor, or a specific operating system such as Microsoft Windows. In such instances, a computer lacking the specific processor or operating system can execute the object code only if it has an emulator program that simulates the necessary processor or operating system or if the code first is run through a translator program that converts it into object code readable by that computer.

Universal City Studios, Inc. v. Reimerdes, 111 F. Supp. 2d 294, 306 (S.D.N.Y. 2000)

The parties agree on IBM's proposed construction, but the intrinsic evidence provides no support for PSI's additional language and it should be rejected.

2. "semantic routine" (claims 1, 4, 9, 12)

The parties' only dispute is whether a "semantic routine" must be limited to a "sequence of instructions" (PSI's construction), or can include a "set or sequence of instructions" (IBM's construction). PSI yet again mischaracterizes IBM's position in two ways. Contrary to PSI's assertion, IBM's construction does not provide for execution in any random order, nor does it allow for "any defined set of instructions." IBM agrees that instructions are not executed in a random order. They are executed in an order appropriate to accomplish the relevant task of the routine, which is to emulate an associated guest instruction. PSI states that IBM's proposed construction allows "an unordered set [of instructions]" (PSI Brief 68), but ignores the essential limitation present in both parties' construction that the semantic routine must emulate an associated guest instruction. If PSI were correct and instructions executed in a random order would not function, then they could not emulate an associated guest instruction. As a result, they would not fall under IBM's proposed construction.

PSI attempts to explain that its construction is not overly-narrow because "there is nothing in the concept of ***sequence*** that ***requires*** one and only one fixed internal order." (PSI Brief 71.) But a fixed order is precisely what sequence means. A "chronological sequence" or "alphabetical sequence" requires one and only one fixed order. (*See* Smotherman Decl. II, ¶ 20.) This very restriction renders the word "sequence" entirely inappropriate to describe a set of instructions that may be executed in one of many different orders.

Furthermore, PSI's reliance on intrinsic evidence (*see* PSI Brief 69) confuses the order in which the instructions of a routine are stored in memory and how they are executed. PSI claims that "everyone in the art knows that computer instructions must be performed as a sequence." (PSI Brief 68). Any argument that instructions are performed in the identical sequence in which they are stored is simply false. Routines often contain conditional branches, instructions that are only executed under certain conditions, loops, subroutines, and other structures that change the order of execution. If PSI means that instructions must be performed in a linear sequence identical to the sequence in which they are stored in memory, then it contradicts the most basic knowledge of programming. (*See* Smotherman Decl. II ¶ 21.)

If, on the other hand, PSI means that instructions are executed in some order, and relies on this execution sequence to define the scope of the term "semantic routine," then IBM's broader construction is more appropriate. Indeed, the very example PSI cites from the '520 patent reveals the problem with its logic. (*See* PSI Brief 69-70.) The instructions "LOAD REG1, MEM1" and "LOAD REG2, MEM2" can be executed in either order with the same result. PSI's expert Dr. Patt wrote:

> Instructions were allowed to execute whenever their resources (data and functional units) became available, ***independent of their order in the program***....

(Ex. 55; *see also*, Smotherman Decl. II, ¶ 21.)

IBM does not dispute that ***parts*** of a semantic routine can contain a sequence.[17] Instead, IBM's construction recognizes that the ***entire*** semantic routine does not have to be a sequence. PSI's "X=wake up, Y=shower, Z=get dressed" example includes three dependent steps. A similar example, X=wake up, Y=shower, Z=eat breakfast (where Y and Z are not dependent on each other), illustrates that this routine can be performed as either X, Y, Z or X, Z, Y. Because

[17] PSI relies on testimony from the inventor of the '520 patent to support its narrow construction. (*See* PSI Brief 70.) However, his quoted testimony, and in particular his statement that a semantic routine's "instructions have to be performed some before the other," is more consistent with IBM's "set or sequence" construction than PSI's narrow "sequence" construction.

instructions are not always dependent on each other (as explained above), IBM's "set or sequence" construction is more appropriate.

In fact, as set forth in IBM's Opening Brief, the intrinsic evidence shows that a semantic routine can be either a "set or sequence," and should not be limited to only a sequence. (IBM Brief 42-43; *see also*, '520 patent, col. 2:9-10; col. 3:1-2; col. 3:28; col. 4:46.)

3. "means, responsive to receipt of said guest memory access instruction for emulation, for translating said guest logical address into a guest real address and for thereafter translating said guest real address into a native physical address" (claim 9)

PSI again misrepresents IBM's recitation of corresponding structure for this claim. IBM is not proposing "every" data processing system, but rather the data processing system described by the '520 specification (Figs. 1, 2, or 3) and equivalents thereof, followed by the recited function and the algorithm for performing the recited function (Fig. 7 and col. 13-52-14:45, Fig. 8 and col. 14:46-15:7, or Fig. 9 and col. 15:56-65).

PSI relies on *Aristocrat/Int'l Game* to argue that IBM's corresponding structure is inadequate, but ignores the fact that in addition to citing a data processing system and structure for that system, IBM also cites the algorithm for performing the recited function. However, IBM actually discloses the algorithm for performing the recited function, which was missing in *Aristocrat/Int'l Game*. In fact, PSI later acknowledges this itself. (PSI Brief 78.).

PSI agrees that Figs. 7, 8, and 9 are corresponding structure, but argues that the portions of the '520 specification that describe those figures are not themselves corresponding structure. PSI has since acknowledged on June 25, 2008 that IBM's construction is correct and now agrees that those parts of the '520 specification are corresponding structure. (*See* Ex. 53.)

4. "means for executing a semantic routine that emulates said guest memory access instruction utilizing said native physical address" (claim 9)

The only remaining issue is whether Figs. 1, 2, 3, and 9 should be included as corresponding structure. IBM's position regarding Figs. 1-3 is fully discussed above for the first

means-plus-function claim element, and equally applies here. IBM is willing to agree that Fig. 9 is part of the corresponding structure for this claim element.

B. The '261 "Emulation" Patent

1. "patching instruction" (claims 1, 2, 7, 8, and 10)

PSI agrees with IBM's construction, but insists on adding limiting language from the specification into the claims, even though the Federal Circuit expressly forbids this. *Phillips v. AWH Corp.,* 415 F.3d 1303, 1323 (Fed. Cir. 2005) (en banc) ("although the specification often describes very specific embodiments of the invention, we have repeatedly warned against confining the claims to those embodiments"). Even worse, the language PSI is attempting to import bears no relationship to the term "patching instruction."

To support this limiting construction, PSI cites a long passage from the SUMMARY OF THE INVENTION section (col. 5:8-28) and then tries to connect this passage to "patching instructions." PSI then insists that because this passage "expressly defines 'patching,' repeatedly states that it is done 'dynamically,' and emphasizes that this dynamic manner is what distinguishes the 'patching' disclosed in the patent from the prior art," it should be part of the construction for "patching instruction." (PSI Brief 83-84.) However, a closer look at the SUMMARY OF THE INVENTION reveals that much of what PSI is saying is simply not true:

- The cited passage does not "expressly define" the term "patching," much less "patching instruction," the actual term in dispute. Indeed, it does not mention either term.
- PSI tries to connect the cited passage to "patching instructions" by stating that it describes "the process of modifying target routines," which the specification later recites is "patching." However, the cited passage nowhere mentions "modifying target routines." Instead, the passage refers to the "dynamic translation of ***incompatible*** instructions," and even PSI concedes in its proposed construction that patching instructions operate on "target instructions," not incompatible ones.

PSI goes on to cite col. 8:50-57 and claim that this "straight-up lexicography" supports its position. However PSI fails to quote the entire passage (portion cited by PSI in bold):

> An emulator mode function 103 processes the target instructions of the selected target translation routine to reflect specifics of the incompatible instruction instance to be emulated by moving any necessary fields from the incompatible instruction to appropriate target instruction(s) in the target routine before each of

> its target instructions is executed. ***This process of modifying the target instructions in a target routine is herein termed "patching."***

('261 patent, col. 8:50-57.) IBM agrees that this passage defines the term "patching." However, it does so without using the terms or even the concepts "dynamically," "in a dynamic manner," or "each time a guest instruction is encountered." It refers only to copying and modifying target instructions. Far from supporting PSI's position, this passage makes clear that IBM's position is correct.

PSI also relies on several Federal Circuit cases, all of which are clearly distinguishable. In *Sinorgchem Co. v. ITC*, 511 F.3d 1132, 1138 (Fed. Cir. 2007), the court upheld the inclusion of a definition from the specification, but stated that "[w]hen the specification explains and defines a term used in the claims, ***without ambiguity or incompleteness***, there is no need to search further for the meaning of the term." *Id.* (emphasis added)(quotes and citation omitted). Neither is the case here.

In *Microsoft Corp. v. Multi-Tech Sys., Inc.*, 357 F.3d 1340, 1348 (Fed. Cir. 2004), the Federal Circuit affirmed a construction limiting the claims to a direct point-to-point telephone line connection. It based its decision on the fact that the specification and prosecution history expressly included this limitation:

> In fact, the specification refers to data transmission "over" or "through" a telephone line ***roughly two dozen times***. Nowhere does it even suggest the use of a packet-switched network. In light of those clear statements in the specification that the invention [] is directed to communications "over a standard telephone line," we cannot read the claims ... to encompass data transmission over a packet-switched network such as the Internet. Instead, the specification shared by all three patents leads to the ***'inescapable conclusion'*** that the communications between the local and remote sites of the claimed inventions must occur directly over a telephone line.

Id. at 1348 (emphasis added); s*ee also Dekalb Genetics Corp. v. Syngenta Seeds, Inc.*, 2007 WL 4564196, at *22 (Fed. Cir. 2007) (distinguishing *Microsoft* because "[t]he present case does not have such specific language in the specification"). Here, the word "dynamically" is only used a few times and is not used once in the Detailed Description section, and there are no "inescapable conclusions" to be drawn from its few uses.

Finally, in *Alloc, Inc. v. ITC*, 342 F.3d 1361, 1368 (Fed. Cir. 2003), the Federal Circuit relied not only on the unambiguous language of the specification to support the claim construction, but on the fact that the patentee relied on the limitation at issue to expressly disavow prior art during prosecution. Here, the '261 inventors never relied on "dynamic" emulation to disavow prior art during prosecution of the application. *See also Johnson & Johnson Vision Care v. Ciba Vision*, 540 F.Supp.2d 1233, 1250 (M.D. Fla. 2008) ("[U]nlike the patent described in *Alloc*, the language of the claims and specifications here do not criticize prior art as lacking the proposed limitations, nor do all embodiments contain the limitations, and thus, the claims and specifications are not 'sufficiently clear' that all claims are limited by the terms 'surface modification' and 'co-continuous phases.'")

Even if everything PSI claimed were true (which it is not), in the end PSI is still attempting to add a new limitation of "dynamic emulation" to the claims, even though the claims themselves recite nothing about dynamic emulation. As noted above, the Federal Circuit expressly forbids importing limitations from the specification, and for this reason alone PSI's construction should be rejected.

2. "incompatible instruction" (claims 1, 7, 8, and 10) and "target instruction" (claims 1, 2, and 7-12)

IBM addresses many of the disputed issues for the terms "incompatible instruction" and "target instruction" in its "instruction" section (*see* Section II, *supra*), and will not repeat those points here.

As set forth in IBM's Opening Brief, its construction is based on the agreed construction for "incompatible" and "target," along with IBM's construction for "instruction." PSI relies on language similar to its manufactured language for "instruction" ("can be directly executed by the processor to which it is directed").

The primary intrinsic evidence on which PSI relies is the preambles for claims 1 and 7 of the '261 patent. The preamble of claim 1 recites "incompatible ... instructions natively executable on a different processor," and "target instructions executable on the target processor."

('261 patent, col. 19:22-29.) The preamble of claim 7 recites "incompatible ... instructions executable on a different processor," and "target instructions executable on the target processor." (Id., col. 19:22-29.) PSI argues that "the claims explicitly say that 'incompatible instructions' are things that can be ***directly*** executed by a processor of the emulated (*i.e.*, incompatible) system, while the target instructions are things than can be ***directly*** executed by the target processor." (PSI Brief 85, emphasis added.) However this is not accurate. Claims 1 and 7 only recite that incompatible instructions are executable on a different processor, not directly executed by a processor of the emulated system.[18] Similarly, those claims only recite that target instructions are executable on the target processor, not directly executed by the processor to which they are directed.

As IBM previously explained in its "instruction" section (*see* Section II, *supra*), PSI has inserted the word "directly" into its various "instruction" constructions without any intrinsic support. PSI presumably takes the "directly executed" language from the IBM Dictionary of Computing's definition of "machine instruction" ("can be directly executed by a processor of a computer"). Then, as it did for "instruction" and "machine instruction," it modifies the definition for "incompatible instruction" and "target instruction" so as to create additional non-infringement arguments. For example, PSI's construction for the term "incompatible instruction" is not "and ***can be*** directly executed by a processor of a computer," but rather "and ***would be able to be*** directly executed by a processor of the emulated processing system." PSI's minor modifications may appear inconsequential, but the difference between whether an instruction "can be directly executed" and "must be directly executed" is substantial.

Finally, regardless of the fact that the '261 specification references machine instructions, the '261 claims themselves never refer to incompatible and target instructions as machine instructions. Therefore PSI's attempt to insert limiting extrinsic evidence into the '261 claims

[18] Claim 1 recites that incompatible instructions are "natively executable on a different processor," meaning that they are executable on the processor of the native computer architecture. ('261 patent, col. 19:24-25.)

should be rejected. *See Phillip v. AWH Corp.*, 415 F.3d 1303, 1323 (Fed. Cir. 2005) (en banc) ("although the specification often describes very specific embodiments of the invention, we have repeatedly warned against confining the claims to those embodiments").

3. "target routine" (claims 1, 2, 7-9, 11, and 15)

The two disputes for "target routine" have long been (1) whether a target routine is a "defined sequence of target instructions" (as PSI maintains) or a "set or sequence of target instructions" (IBM's construction), and (2) whether a target routine "performs a function similar to, but not necessarily identically to, a corresponding incompatible instruction" (PSI) or whether it "enable[s] an incompatible instruction to be run on [a] target computer system" (IBM).

On June 25, 2008, two days before IBM's Reply Brief due date, PSI changed its construction to just the phrase "a defined sequence of target instructions." (Ex. 53.) Based on this new construction, IBM is willing to change its construction to "a set or sequence of target instructions," and the only remaining dispute is whether a target routine is a "set or sequence of target instructions" or a "defined sequence of target instructions."

The parties similarly dispute whether a "semantic routine" in the '520 patent is a "sequence of instructions" or a "set or sequence of instructions," and many of IBM's arguments on that dispute apply here as well. (*See* Section IV.A.2, *supra*.) In particular, PSI's statements that a target routine cannot be executed "in any order" are irrelevant. If a target routine executed in any order would not work, then it cannot possibly satisfy IBM's construction requiring the routine to "enable an incompatible instruction to be run on a target computer system." A routine that – as PSI characterizes it – would not work would also not enable the incompatible instruction to be run.

PSI relies on claims 7 and 18 to support its construction. (PSI Brief 88-89.) While those claims do recite executing the target routines in a "sequence" (claim 7) or "execution sequence" (claim 18), independent claim 1 does ***not*** limit a target routine to a sequence. ('261 patent, col. 20:6-11.) Because it is contrary to the laws of claim construction to import a limitation from one independent claim to a separate independent claim, the Court should reject PSI's limiting

construction. *See Modine Mfg. Co. v. U.S. Intern. Trade Com'n,* 75 F.3d 1545, 1551 (Fed. Cir. 1996), *abrogated on other grounds by Festo Corp. v. Shoketsu Kinzoku Kogyo Kabushiki Co.*, 234 F.3d 558 (Fed. Cir. 2000), *rev'd by* 535 U.S. 722 (2002) ("'Where, as here, the limitation sought to be 'read into' a claim already appears in another claim, the rule is far more than 'general.' It is fixed. It is long and well established. It enjoys an immutable and universally applicable status comparatively rare among rules of law. Without it, the entire statutory and regulatory structure governing the drafting, submission, examination, allowance, and enforceability of claims would crumble.' When a limitation is included in several claims but is stated in terms of apparently different scope, there is a presumption that a difference in scope is intended and is real.") (citation omitted); *see also C.R. Bard, Inc. v. M3 Systems, Inc.*, 157 F.3d 1340, 1362 (Fed. Cir. 1998) ("improper to import limitation from one claim into another claim lacking the limitation") (*citing D.M.I., Inc. v. Deere & Co.*, 755 F.2d 1570, 1574 (Fed. Cir. 1985)). For this reason alone, IBM's "set or sequence" construction is more appropriate.

The word "sequence" appears only once in the specification, describing prior art U.S. Patent No. 4,587,612: "That patent teaches the substitution of specific register values, and immediate displacement values, from source machine instruction into target machine instructions which, either singly or as ***a sequence of more than one***, perform the function of the source machine. ***The technique described cannot be used to provide a total operational S/390 program execution environment***." That is, while the inventors viewed the prior art as employing a sequence of instructions, that prior art was insufficient to solve the problem addressed by the '261 patent.

The '261 patent does not use the word "sequence" again in the specification. To impose the limitation "sequence" on the definition of "target routine" would be confusing and unfairly limiting and should be rejected. PSI quotes the specification: "each incompatible instruction instance is translated to ***a <u>set</u> of equivalent target instructions in a corresponding target translation routine*** for execution in the target processor." PSI claims that "this proves nothing" because "set" is broader than "sequence." (PSI Brief 89.) Indeed, it is a broader term, and IBM

is entitled to the broad construction precisely because the inventors used the term "set" to describe their invention, but only used the narrower term "sequence" when referring to prior art.

PSI mischaracterizes the "sole inventive portion of the patent" in the Notice of Allowability. (PSI Brief 89.) The full quote reads:

> the prior art of record does not teach associating one or more patching instructions with a target instruction in a target routine when the target instruction requires modification for enabling the target routine to provide execution results identically to execution results defined for the corresponding incompatible instruction by the incompatible architecture and not associating any patching instruction with any target not requiring modification.

(PSI Ex. 27, 8/02/99 Notice of Allowability, p. 2.) This statement implies nothing about the nature of "set" or "sequence." PSI also points to testimony from the inventor of the '520 patent, even though "target routine" never appears in the '520 patent.

IBM's construction is thus the proper one and should be adopted.

V. THE FLOATING POINT PATENTS

A. The '709 "Rounding Mode" Patent

The only terms in dispute, "processor" and "instruction," are addressed in Section II.

B. The '678 "Data Class" Patent

1. "a floating point processor" (claim 1)

The parties' constructions are markedly different. IBM's construction is based on its construction for the term "processor," while PSI's construction identifies one possible component of a floating point processor and a purported function of that component. IBM has already addressed several of the disputed issues for a "floating point processor" in its "processor" section, and will not repeat those points here. (*See* Section II, *supra*.)

A closer look at the intrinsic evidence on which PSI relies reveals the inadequacy of its limiting construction. PSI first concedes that a floating point processor "executes floating point instructions" (IBM's construction), then points to Fig. 2 of the '678 patent and a citation from the '678 specification:

Instruction unit **200** fetches instructions from common main storage **110** according to an instruction address located in the program status word (PSW) register **202**, and appropriately effects execution of these instructions. Instruction unit **200** appropriately hands off retrieved floating point instructions to floating point unit **204**, along with some of the operands that may be required by the floating point unit to execute the instruction. Floating point (FP) unit **204** includes all necessary hardware to execute the floating point instruction set, and preferably, in accordance with an embodiment of the present invention, supports both Binary and Hexadecimal floating point formats. FP unit **204** is coupled to floating point (FP) registers **206**, which contain floating point operands and results associated with FP unit **204** processing, and is also coupled to general registers **208**. FP unit **204** is also coupled to floating point control (FPC) register **210**, which preferably includes mask bits in addition to those provided in the PSW. In a multi-user application, FPC register **210** is under control of the problem state program.

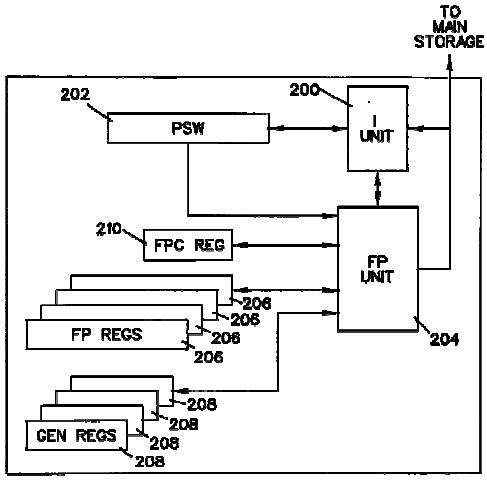

FIG. 2

PSI relies on the embodiment described above to argue that (1) a floating point unit must be part of the floating point processor, and (2) the floating point unit is what executes floating point instructions. (PSI Brief 112-113.) The intrinsic evidence does not support these conclusions.

First, the embodiment of Fig. 2 illustrates several components that make up the processor, including the Instruction Unit 200, Program Status Word 202, Floating Point Unit 204, Floating Point Registers 206, General Register 208, FPC Registers 210, and presumably other components as well (since Fig. 2 is only a block diagram). The '678 specification also describes these components, and indicates that their interaction allows the processor to interpret and execute floating point instructions.

PSI's construction includes the floating point unit ***only***, and ignores the other components of the processor. PSI's construction is also based solely on this single embodiment described in the '678 specification, whereas nothing in the '678 claims limits a floating point processor to a floating point unit. In fact, the term "floating point unit" appears nowhere in the '678 claims.

IBM's construction, on the other hand, does not limit the term floating point processor to a single embodiment, but instead includes the common thread for all floating point processors: interpreting and executing floating point instructions. The IBM Principles of Operation, which

were incorporated by reference into the '678 specification, directly address this point for processors (*i.e.*, CPUs):

> The physical implementation of the CPU may differ among models, but the logical function remains the same. The result of executing an instruction is the same for each model, provided that the program complies with the compatibility rules.

(Ex. 25.) The point of this statement is that any processor, including a floating point processor, can be implemented in different ways, but the function of interpreting and executing instructions remains the same regardless of implementation.

PSI's extrinsic evidence provides no justification for limiting the floating point processor to the floating point unit. The *IBM Journal of Research and Development* article says nothing about the floating point processor being a floating point unit, and at best it is one of many possible implementations of a floating point processor. The 1997 Schwarz article is two years after the 1995 filing date of the '678 patent and bears no relation to that patent. U.S. Patent No. 7,117,389 was filed over eight years after the 1995 filing date of the '678 patent, and also bears no relation to the '678 patent. Finally, the IBM-Intel license is unrelated to the '678 patent, and makes no reference to a floating point processor. In addition, the 1989 IBM-Intel License Agreement expressly covers IBM patents "issued or issuing on patent applications entitled to an effective filing date prior to January 1, 1994." (PSI's Ex. 10 at 10.) The '678 patent's effective filing date is March 31, 1995, after the January 1, 1994 deadline.

IBM's construction is more appropriate and should be adopted.

2. "a machine instruction" (claims 1, 3, 7, 9)

The only remaining dispute is whether a machine instruction "can be directly executed by a processor of a computer" (IBM's construction taken ***verbatim*** from the IBM Dictionary of Computing), or "can be directly executed by the processor to which it is directed" (PSI's construction created without any intrinsic or extrinsic support). IBM has already addressed most of PSI's arguments (*see* IBM Opening Brief 54-55; *see also* Section II, *supra*), and will not repeat those points here.

PSI's opening point on "machine instruction" is that IBM relies only on extrinsic evidence to support its construction and to rebut PSI's construction. But PSI itself offers no relevant intrinsic evidence, and contorts its limited extrinsic evidence to match its contrived construction. For example, PSI cites multiple passages from the '678 specification (along with attorney argument) to support its construction, yet those passages say ***nothing*** about a "machine instruction," "***direct*** execution," or "machine instructions being directly executed by the processor to which it is directed." (*See* PSI Brief 117.)

PSI also misrepresents IBM's construction as something that "can be 'directly executed' by any processor whatsoever," or "***any*** processor of ***any*** computer interpretation." (PSI Brief 116, 117.) But this is plainly misleading. The IBM Dictionary of Computing defines "machine instruction" as "an instruction that can be directly executed by a processor of a computer." (Ex. 29.) IBM's construction uses these words ***verbatim***: "A string of digits that specifies an operation and identifies its operands, if any, and ***can be directly executed by a processor of a computer***." A machine instruction that "can be directly executed by a processor of a computer" means exactly what it says, namely, that there is at least one processor which can directly execute that particular instruction. This is far different from saying that a machine instruction can be directly executed by "any processor whatsoever," or "***any*** processor of ***any*** computer interpretation."

PSI similarly misrepresents IBM's construction when it states that "[u]nder IBM's construction, the System-z 'Test FP Data Class' instruction would be a 'machine instruction' in an Intel-based machine even though the Intel processor treats it only as data and cannot execute it." (PSI Brief 116.) In fact, assuming the Test FP Data Class instruction is in the form of a string of digits (*i.e.*, at the machine code level), it is a machine instruction because it can be directly executed by a processor of a computer (*e.g.*, IBM's processors).

PSI provides no extrinsic evidence to support its construction, apart from the IBM Dictionary of Computing's definition of "machine instruction." However, IBM adopts this definition ***verbatim*** and, accordingly, its construction should be adopted.

3. "means for retrieving the floating point number from memory" (claim 1)

PSI relies on *Aristocrat/Int'l Game* to argue that IBM's corresponding structure is inadequate, but ignores that IBM cites a shared memory computer system, structure for that system, and the algorithm for performing the recited function. In *Aristocrat/Int'l Game*, the patentee only cited "a standard microprocessor-based gaming machine with appropriate programming." *Aristocrat/Int'l Game*, 521 F.3d at 1333. IBM's disclosure goes far beyond what was disclosed in *Aristocrat/Int'l Game*, and provides an algorithm sufficient for one of ordinary skill to implement the recited function.

PSI also alleges two problems with IBM's recited algorithm (illustrated below) for "retrieving the floating point number from memory":

> The floating point number in the floating point register that is pointed to by R1 is loaded into the convert to type number logic and a determination is made as to the data class of the number.

('678 patent, col. 3:63-66; *see also* Fig. 8.) First, PSI states that "by the time the floating point number is 'in the floating point register,' it has already been retrieved from memory." (PSI Brief 121.) PSI fails to recognize that a register *is* memory. The recited algorithm plainly states that the floating point number in the floating point register, which is memory, is "loaded in the convert to type number logic." A person of ordinary skill at the time of the invention would understand that for the floating point number to be loaded, it must first be ***retrieved*** from the register, which PSI admits is memory. (*See* PSI Brief 30.) Indeed, PSI's own definition of register acknowledges that it is a "storage element" (PSI Brief 25), which of course is synonymous with memory.

Second, PSI states that Fig. 8 and the corresponding language simply describe the function to be performed, and therefore are insufficient structure under *Aristocrat/Int'l Game*. (PSI Brief 121.) But PSI ignores the Federal Circuit's holding that "the sufficiency of the disclosure of algorithmic structure must be judged in light of what one of ordinary skill in the art would understand the disclosure to impart." *Aristocrat/Int'l Game*, 521 F.3d at 1337. The

Federal Circuit further held that the patentee "was not required to produce a listing of source code or a highly detailed description of the algorithm to be used to achieve the claimed functions in order to satisfy 35 U.S.C. 112 ¶ 6." *Id.* Here, it is hard to fathom how a person of ordinary skill would not understand how to retrieve a number from memory.

PSI's attempt to invalidate the '678 claims that include this element (claims 1 and 6) must be rejected, and IBM's construction should be adopted.

4. "means for determining whether the data class of the floating point number is the identified data class by examination of condition of the fields of the floating point number" (claim 1)

The parties agree that Fig. 9 of the '678 patent is corresponding structure. The remaining issues for this claim element are whether: (1) Figs. 1 and 2 and the written description accompanying those figures (col. 2:35-38 and col. 2:39-59) are corresponding structure; (2) Figs. 5, 7, and 8 and the written descriptions accompanying those figures are corresponding structure; (3) the written description accompanying Fig. 9 is also part of the corresponding structure; and (4) IBM's proposed structure is linked to the recited function.

As previously explained in the context of other means-plus-function elements, the "shared memory computer system" described by Figs. 1 and 2 performs the recited function, much like the special purpose computer performing the disclosed algorithm in *WMS Gaming, Inc. v. Int'l Game Tech.*, 184 F.3d 1339, 1349 (Fed. Cir. 1999) ("In a means-plus-function claim in which the disclosed structure is a computer, or microprocessor, programmed to carry out an algorithm, the disclosed structure is not the general purpose computer, but rather the special purpose computer programmed to perform the disclosed algorithm.").

PSI argues that the corresponding structure is a portion of Fig. 9 only, and not Figs. 5, 7, and 8 and the written descriptions accompanying those figures. (PSI Brief 122-123.) But those figures and the written descriptions accompanying them are directly related to Fig. 9, and necessary for a person of ordinary skill to understand Fig. 9. For example, Fig. 5 illustrates the location of the Fig. 9 "12-BIT MASK" in the instruction. Fig. 7 illustrates the 12 floating point data classes associated with the Fig. 9 "12-BIT MASK." Fig. 8 illustrates the "BIT

SELECTION" for the Fig. 9 "12-BIT MASK." And the cited portions of the '678 specification are written descriptions accompanying those figures, and should be included with them as corresponding structure. *See Texas Digital Systems, Inc. v. Telegenix,* 308 F.3d 1193,1212 (Fed. Cir. 2002) ("The description in the specification of the structure corresponding to the recited function is not limited to Figures 3 and 4, as instructed, but also includes the written description accompanying these Figures").

The parties also dispute whether the written description accompanying Fig. 9 (col. 4:9-32) should be included as corresponding structure. This description specifically explains the block diagram in Fig. 9, including many of the "short-hand" symbols (*e.g.*, wz, en, eo, fz, fn) which might not be readily apparent to a person of ordinary skill. Moreover, the Federal Circuit has made clear that written description accompanying a figure is part of the corresponding structure. *See Texas Digital Systems*, 308 F.3d at 1212.

Finally, PSI relies on *B. Braun* to argue that IBM's proposed structure (other than a portion of Fig. 9) is not clearly linked to the recited function. (PSI Brief 122-123.) But the facts here are clearly distinguishable from those in *B. Braun*, where the Federal Circuit found no indication that the structure proposed by the patentee was connected to the recited function. *B. Braun Med., Inc. v. Abbott Labs,*, 124 F.3d 1419, 1425 (Fed. Cir. 1997) ("Although Fig. 3 of the patent shows a valve seat, neither the specification nor the prosecution history contains *any* indication that the valve seat structure corresponds to the recited function.") (emphasis in original). Here, in addition to the undisputed linking language for Fig. 9, the other corresponding structure proposed by IBM is directly related to Fig. 9, and a person of ordinary skill would have understood the connection to the recited function. Further, the Federal Circuit has established that the standard for establishing a link between the recited function and its corresponding structure is low. *Medical Instrumentation*, 344 F.3d at 1213-14 ("[W]e have been generous in finding something to be a corresponding structure when the specification contained a generic reference to structure that would be known to those in the art and that structure was clearly associated with performance of the claimed function.")

For these reasons, IBM's proposed corresponding structure should be adopted.

5. "means for setting a condition code in a program status word based upon the determination of whether the data class is the identified data class" (claim 1)

The parties agree that Fig. 9 is corresponding structure. Similar to the previous claim element, the remaining issues are whether: (1) Figs. 1 and 2 and the written description accompanying those figures (col. 2:35-38 and col. 2:39-59) are corresponding structure; (2) Fig. 8 and the accompanying written description are corresponding structure; (3) the written description accompanying Fig. 9 is also part of the corresponding structure; and (4) IBM's proposed structure is linked to the recited function.

IBM already addressed issue (1) for the previous claim element.

Regarding issue (2), Fig. 8 and the short passage describing that figure ('678 patent, col. 4:5-8) are clearly related to Fig. 9 (which PSI agrees is corresponding structure). PSI cites this passage as "linking language." (PSI Brief 126.) Also, the '678 specification states that "Fig. 9 illustrates the conversion circuitry of the convert to type number logic." ('678 patent, col. 4:9-10.) This "convert to type number logic" is one of the referenced blocks in the Fig. 8 block diagram:

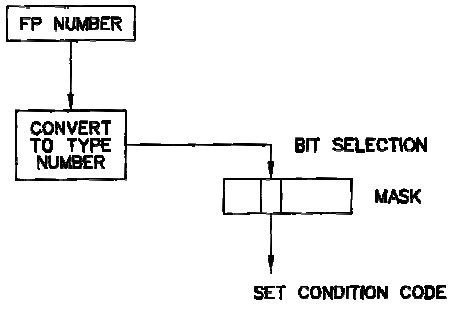

FIG.8

Fig. 8 and col. 4:5-8 are undeniably related to Fig. 9, providing the context necessary to a person of ordinary skill, and should therefore be included as corresponding structure.

For issue (3), PSI first argues that col. 4:9-32 should not be part of the corresponding structure, but then recites this passage as "linking language." (PSI Brief 126.) PSI goes on to state in bold that "IBM does not identify any linking language in the patent." *Id.* IBM can only

surmise that PSI believes that "linking language" is somehow separate and distinct from corresponding structure, and that IBM's recitation of this passage as corresponding structure means it cannot be linking language. But this is at odds with both *B. Braun* and *Texas Digital*. In *B. Braun*, the Federal Circuit did not distinguish linking language from corresponding structure, and in fact found that 6 lines in the patent at issue both described the structure and provided adequate linking language. *B. Braun*, 124 F.3d at 1424. In *Texas Digital*, the Federal Circuit determined that language in the specification describing a figure is corresponding structure along with that figure. *Texas Digital Systems*, 308 F.3d at 1212. Thus, PSI is wrong to argue that linking language cannot be corresponding structure, as it is here.

For these reasons, IBM's construction is more appropriate and should be adopted.

C. The '106 "Floating Point Conversion" Patent

1. "floating point unit" (claims 11 and 12)

To simplify this issue, IBM is willing to agree that the floating point unit is the portion of the processor that performs floating point operations. Thus, the only dispute remaining is whether the floating point unit is part of the "processor" (IBM's construction) or part of the "processor's circuitry" (PSI's construction).

PSI frames its arguments to suggest that IBM is disputing the presence of any hardware for the floating point unit. To the contrary, IBM is not disputing that the floating point unit can be implemented in hardware, but is instead proposing a construction in which the floating point unit can be implemented in hardware or a combination of hardware and software. PSI itself admits that the '106 specification describes one embodiment of a floating point unit performing floating point "operations performed by a software routine which operates on the 3 portions of data." (PSI Brief 108; *see* '106 patent, col. 7:57-59.) This intrinsic evidence completely contradicts PSI's position that the floating point unit is limited to hardware.

The other intrinsic evidence on which PSI relies does not support its hardware-only construction either. For example, PSI cites the following passage from the '106 specification:

> There is a need, therefore, to have a computer system that supports two floating architectures. Moreover, such a computer system should provide the two floating

> point architectures with high performance and with limited resources. That is, multiple floating point architecture support should be provided without having to essentially duplicate floating point hardware and without sacrificing performance.

('106 patent, col. 1:44-50.) PSI emphasizes that "multiple floating point architecture support should be provided without having to essentially duplicate floating point hardware and without sacrificing performance." (PSI Brief 96.) But this says nothing about whether the invention (or the floating point unit, for that matter) must be implemented in hardware alone, and only describes how the invention is ***not*** implemented. In fact, this language suggests that the goal of the invention is to reduce reliance on hardware, which can be accomplished by combining hardware and software. A person of ordinary skill at the time of the invention would have been aware of such hardware/software floating point implementations. (*See* Smotherman Decl. II, ¶¶ 5, 19.)

PSI also points to Fig. 1 and asserts that it "clearly depicts hardware circuitry." (PSI Brief 98.) Contrary to this assertion, Fig. 1 does not "clearly depict hardware circuitry," but instead represents a block diagram illustrating "an overview of the dataflow of a Floating Point Unit (FPU)." ('106 patent, col. 2:14-15.)

All of PSI's purported extrinsic evidence is dated well after the 1995 filing date of the '106 patent:

- Comprehensive Dictionary of Electrical Engineering: Copyright 2005
- Microsoft Computer Dictionary: Copyright 2002[19]
- IBM web site, dated 2008
- Schwarz Article, dated 1997
- U.S. Patent No. 7,117,389, filed September 18, 2003

As such, none of it is helpful in determining the term's meaning to one of skill in the art at the time of the invention. *See Genzyme Corp. v. Transkaryotic Therapies, Inc.*, 346 F.3d 1094, 1098 (Fed. Cir. 2003) ("A fundamental principle for discerning a term's usage is the ordinary and

accustomed meaning of the words amongst artisans of ordinary skill in the relevant art at the time of invention.").

PSI's hardware only construction is not supported by the intrinsic or extrinsic evidence and should be rejected.

2. "a floating point unit having an internal dataflow" (claim 11)

In PSI's Brief, it agreed to IBM's proposed language of "executing floating point operations" (instead of "perform[ing] floating point calculations.") On June 25, 2008, PSI agreed to accept IBM's construction in its entirety. (Ex. 53.)

3. "(a floating point unit that) supports both said first floating point architecture and said second floating point architecture" (claim 11)

PSI agrees with IBM's construction but asserts that its own construction is more appropriate. (PSI Brief 103.) However, PSI's construction, and in particular its use of the language "on its own," is not supported by any intrinsic or extrinsic evidence.

PSI's proposed construction is that a floating point unit "supports" a floating point architecture when it can, ***on its own***, perform calculations on floating point numbers of that architecture. Thus, according to PSI's construction, a floating point unit would have to be able to, on its own, perform calculations for each of the two floating point architectures. PSI offers this single piece of intrinsic evidence (from the '106 specification) to support its construction:

> There is a need, therefore, to have a computer system that supports two floating architectures. Moreover, such a computer system should provide the two floating point architectures with high performance and with limited resources. That is, multiple floating point architecture support should be provided without having to essentially duplicate floating point hardware and without sacrificing performance.

('106 patent, col. 1:44-50.) But this says nothing about the floating point unit performing floating point operations for each floating point architecture "on its own," and only identifies an unsatisfactory way to support two floating point architectures.

[19] The 1994 Microsoft Computer Dictionary, which is more appropriate in light of the '106 patent's 1995 filing date, does not include a definition for "floating point unit."

PSI professes it cannot understand why IBM is unwilling to accept the "on its own" language. (PSI Brief 103-104.) The reason is obvious – it has no intrinsic or extrinsic support. And PSI's ulterior motives are also apparent – if the Court agrees with PSI's construction, PSI will then make the non-infringement argument that the Itanium floating point unit in PSI's system cannot "on its own" (*i.e.*, without PSI's software) support two floating point architectures.

For these reasons (and for the reasons set forth in IBM's Opening Brief), the Court should adopt IBM's construction.

4. "converter" (claim 11)

The remaining disputes for the term "converter" are (1) whether a converter is hardware only, and (2) whether each converter must be located within the floating point unit.

First, to simplify the issues , IBM is willing to revise its construction to "hardware, or a combination of hardware and software, that changes data from one format or architecture type to another format or architecture type."[20]

As it does for several other disputed terms and phrases, PSI attempts to suggest that IBM's proposed construction "can, in ***the context of a computer system***, mean literally anything." (PSI Brief 104, emphasis added.) But IBM's construction is not only in the context of a computer system, but also in the context of claim 11. *See Phillips v. AWH Corp.,* 415 F.3d 1303, 1314 (Fed. Cir. 2005) (en banc) ("the context in which a term is used in the asserted claims can be highly instructive"). For example, each of the four "converter" elements in claim 11 recite limitations that provide context (*e.g.*, "a first converter that applies a transformation to convert an operand of said first floating point architecture type to said internal floating point format, and applies said transformation to an operand of said second floating point architecture type to provide transformed second floating point architecture type data.") ('106 patent, col.

[20] PSI posits that "if claim 11 fails to recite any structure [then it] should be construed to be a means-plus-function claim." (PSI Brief 106.) Because IBM's construction includes hardware (either alone or in combination with software), PSI's means-plus-function argument should be rejected.

22:18-23)). PSI's attempt to portray IBM's construction of "converter" as existing in a vacuum should be rejected.

IBM does not dispute that a converter can include hardware, but rather disputes PSI's unsupported position that a converter must be hardware only (*i.e.*, circuitry). While PSI points to various parts of the '106 specification identifying hardware embodiments (many of which were addressed in IBM's Opening Brief), PSI points to no intrinsic evidence that limits a converter exclusively to hardware. For example, PSI identifies Fig. 1 of the '106 patent and points to six converters that it claims are hardware. However, Fig. 1 does not depict hardware. It is a block diagram illustrating "an overview of the dataflow of the Floating Point Unit (FPU)." ('106 patent, col. 2:14-15.) While the '106 specification demonstrates that a converter ***can*** include hardware, it does not demonstrate that a converter can ***only*** be implemented in hardware.

PSI also misuses the '106 prosecution history to suggest that the Examiner's attempt to identify some corresponding structure for the means-plus-function element of claim 1 ("conversion means") should limit the scope of "converter" in claim 11. As the language below makes clear, the Examiner's identification applies only to claim 1:

> The prior art of record does not fairly suggest in a computer system a conversion means as recited in claims 1, and the first through forth converter as recited in claim 11 wherein, in accordance with 35 USC 112, 6th paragraph, the conversion means is construed to cover the corresponding structure of the combination of converters 11, 12, 15, 16, 22, 24, multiplexers 28, 28, 29 and register 13, 14, 14 connected as disclosed in figure 1 or equivalents thereof, and the first and forth converter apply the same transformation for both converting an operand of a first floating point architecture type to the internal floating point format, and transforming an operand of a second floating point architecture type as recited in claim 11.

(Ex. 56, 4/11/97 Notice of Allowability, p. 2.) Also, a Patent Examiner's recitation of corresponding structure pursuant to 35 U.S.C. § 112 ¶ 6 in a Notice of Allowability is not controlling. *See Salazar v. Procter & Gamble Co.*, 414 F.3d 1342, 1344-45 (Fed. Cir. 2005) (holding that an examiner's unilateral statements in a Notice of Allowance did not limit claim scope); *see also Schwing GmbH v. Putzmeister Aktiengesellshaft*, 305 F.3d 1318, 1324-25 (Fed.

Cir. 2002) ("Prosecution history... cannot be used to limit that scope of a claim unless the applicant took a position before the PTO.").

Next, to support its argument that a converter must be located within the floating point unit, PSI asserts that "in the prosecution history the '106 inventors defined 'floating point operation' to include 'conversion' when they distinguished their invention over the prior art." (PSI Brief 107.) PSI directly follows that assertion with two pages of argument in which it attempts to discredit IBM's recitation of software routines in the '106 specification as "misleading," because allegedly these software routines relate to "operations" and not "conversions." (PSI Brief 107-109.) But PSI admits that floating point operations ***include*** conversions – it is clearly talking out of both sides of its mouth.

Finally, as explained in IBM's Opening Brief, claim 11 recites the "floating point unit" and each of the four "converters" as separate elements, with no language limiting the location of any of the converters within the floating point unit. Because the "converters" are recited as separate claim elements from the "floating point unit," each of those converters need not be "within the floating point unit," as would be required under PSI's construction. PSI tries to discredit this position by arguing that IBM does not cite any supporting authority and that none exists. (PSI Brief 109). To the contrary, the Federal Circuit has long held that the words of the claim carry the most weight, as they do here. *See Phillips*, 415 F.3d at 1312 ("It is a 'bedrock principle' of patent law that 'the claims of a patent define the invention to which the patentee is entitled the right to exclude.'") (citation omitted).

VI. THE PARTITIONING PATENTS

A. The '812 "Partition Communication" Patent

1. "host-network interface" (claims 1, 11)

PSI contends that a "host-network interface" must be a "hardware component" (PSI Brief 128), but states that "the host-network interface is a discrete hardware component that is physically connected to the network port and (through a combination of physical circuits and ***software stored in a memory within it***) performs the tasks needed to send and receive signals

coming and going directly from and to the network port." (PSI Brief 131, emphasis added). A host-network interface is not a "discrete hardware component" if it includes software. PSI tries to justify its contradictory position by arguing that a person of ordinary skill at the time of the invention would think of a host-network interface as hardware even if it includes software, and therefore it should be construed as hardware. But this is illogical and unsupported by the intrinsic evidence.

PSI implies that IBM disputes the possible inclusion of hardware as part of the host-network interface (which it does not), and improperly concludes that references in the '812 specification and prosecution history demonstrate a hardware-only implementation. For example, PSI refers to Fig. 4 of the '438 patent and contends that it represents the host-network interface as a "hardware component," when in fact the '438 patent itself describes Fig. 4 as "a block diagram of a host network interface pursuant to an embodiment of the present invention." ('438 patent, col. 4:16-17.) PSI also refers to the "preferred embodiment IBM model 3172 host-network interfaces ... [as] indisputably hardware devices." (PSI Brief 129-130.) To the contrary, the '812 patent does not describe the IBM model 3172 Interconnect Controller as a "preferred embodiment," but in fact expressly refers to it as prior art.[21] ('812 patent, col. 2:1-36, Fig. 1.) PSI fails to mention other intrinsic evidence describing the 3172:

> Although the host to network interface 51 of FIG. 2 is part of the processor 11, other embodiments of the present invention are possible. ***As one example, an IBM 3172 67 (FIG. 3) may be loaded with software to enable its use as the HNI of the present invention; the techniques implemented by the software are discussed further hereinbelow.***

('438 patent, col. 4:52-57, emphasis added.) This directly contradicts PSI's position that the 3172 is "indisputably" a hardware device.

PSI's purported extrinsic evidence refers only to the Open Systems Adapter (OSA) card, which is one part of one embodiment of the host-network interface set forth in the '812 specification. PSI ignores the host channel connection, and the fact that the OSA Adapter

expressly illustrates software in the form of LAN DEVICE DRIVERS. PSI does not completely ignore Fig. 3 of the '812 patent, which shows LAN DEVICE DRIVERS (*e.g.*, software) as part of a host-network interface embodiment, but misrepresents that figure by saying that the OSA Adapter only "utilizes" this software. (PSI Brief 131.) But Fig. 3 (illustrated below) shows that the OSA Adapter ***includes*** this software, as opposed to simply utilizing it.

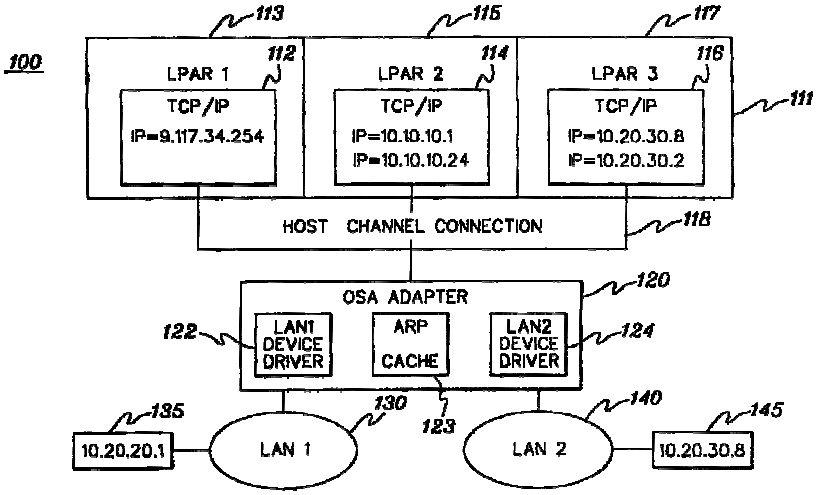

fig. 3

PSI also misrepresents IBM's construction as "unbounded." (PSI Brief 131-133.) As its name implies, the host-network interface is ***between*** the host computer and the network. However, to obviate this dispute, IBM is willing to agree that a host-network interface is "a communication interface coupled between a mainframe class data processing system and a port to a network."

2. "saving at said host-network interface" (claim 1)

PSI insists on construing the phrase "saving at said host-network interface" so that it can add limiting language from the specification, namely, that an internet protocol (IP) address is saved "on memory located within" the host-network interface. A closer look at the parties'

[21] PSI includes Fig. 1 but crops out the heading which identifies it as "PRIOR ART." (PSI Brief 130.)

proposed constructions reveals that the dispute is not about the word "saving" or "host-network interface," but in fact centers on the word "at":

IBM's Construction	PSI's Construction
saving...	saving...
...at...	*...on memory located within...*
...said host-network interface.	...the host-network interface.

IBM's proposed construction for "at" is "at," while PSI proposes that "at" means "on memory located within." PSI cites *O2 Micro Int'l Ltd. v. Beyond Innovation Tech. Co.,* 521 F.3d 1351 (Fed. Cir. 2008) for the proposition that IBM's construction is inadequate. (PSI Brief 133.) Contrary to the *O2 Micro Int'l Ltd.* case, IBM is not asking the Court to avoid construing this phrase, but instead to not limit the construction to "saving ***on memory located within*** said host-network interface."

Although PSI's construction is consistent with one embodiment in the '812 specification, it appears nowhere in the '812 claims. Inserting PSI's proposed language would use the specification to limit the scope of the claims, which the Federal Circuit expressly forbids. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1323 (Fed. Cir. 2005) (en banc).

PSI also relies on dependent claim 2 ("saving each IP address ... in an address resolution protocol (ARP) cache at said host-network interface") to support its construction. (PSI Brief 134.) But claim 2 (which depends on claim 1) recites a narrower limitation for the "saving" step, and "saving at said host-network interface" from claim 1 must be broader in scope. *See* 35 U.S.C. § 112 ¶ 4; 37 C.F.R. § 1.75(c) ("One or more claims may be presented in dependent form, referring back to and further limiting another claim or claims in the same application.... Claims in dependant form shall be construed to include all the limitations of the claim incorporated by reference into the dependent claim.").

A person of ordinary skill at the time of the invention would understand "at" to mean "at." However, if the Court does decide to substitute a different word or phrase to define "at," IBM proposes "on" ("saving ***on*** said host-network interface").

3. "at least one computer usable medium having computer readable program code means embodied therein for causing network communications in a mainframe class data processing system having multiple partitions and a port to a network, the computer readable program code means in the article of manufacture comprising" (claim 11)

PSI misstates IBM's recitation of corresponding structure for this claim. IBM is not proposing "any" article of manufacture, but rather the article of manufacture set forth at col. 10:46-57 of the '812 specification and equivalents thereof, followed by the recited function and, in parentheses, the algorithm for performing the recited function through software on that article of manufacture.

The '812 specification describes the structure of the article of manufacture at col. 10:46-57. Regardless of whether this language appears in other IBM patents, it recites structure for the article of manufacture in this patent. A person of ordinary skill at the time of the invention would certainly understand the structure for a "computer useable media" or "program storage device readable by a machine" that stores at least one computer program of instructions executable by that machine (*i.e.*, software) for performing the recited functions.

PSI's argument that col. 10:46-57 "is so general that it describes all computer programs" is misleading. (PSI Brief 137.) IBM's proposed corresponding structure includes this recitation of structure, ***along*** with the algorithm for the software on that structure that performs the recited function. PSI ignores the second part of IBM's construction when addressing the first part, and vice-versa.

Further, PSI's reliance on *Medical Instrumentation* is inapposite. The Court there determined that a step described only in a box depicted as a method of the invention, and not linked at all to the structure of the apparatus, could not be considered corresponding structure. *Medical Instrumentation & Diagnostics Corp. v. Elektra AB,* 344 F.3d 1205, 1213 (Fed. Cir.

2003). There was no evidence that the figure describing the method step "would be understood by one skilled in the art to refer to structure at all, let alone software for [the claimed function.]" *Id.* at 1213. Furthermore, "had the box ... actually referred to some structure instead of simply referring to a step in the claimed method, then the situation may well be different. In past cases, we have been generous in finding something to be a corresponding structure when the specification contained a generic reference to structure that would be known to those in the art and that structure was clearly associated with performance of the claimed function." *Id.* at 1213-14.

Here, one skilled in the art would recognize IBM's proposed corresponding structure, which includes the algorithm that performs the recited function, and the structure associated with performing the claimed function.

(a) "computer readable program code means for causing a computer to effect saving at a host-network interface an internet protocol (IP) address of at least one of the multiple partitions of the mainframe class data processing system" (claim 11)

IBM cited the following from the '812 specification along with Fig. 5 as the algorithm for performing the function of saving an IP address of one of the partitions at the host-network interface:

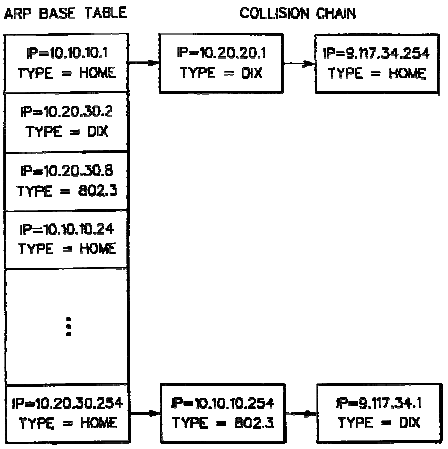

fig. 5

At initialization time, each Host TCP/IP stack is configured to register its HOME IP addresses with the OSA adapter in a manner apparent to one skilled in the art. The "HOME" addresses are those which are recognized as local IP addresses by the specific stack.

PSI claims that this disclosure is inadequate. The '812 specification provides a detailed example of how to save an IP address at the host-network interface. PSI argues that more detail is required, but misconstrues *Aristocrat/Int'l Game* by ignoring the fact that the test for sufficient detail is premised on an interpretation by one of ordinary skill at the time of the invention. *Aristocrat/Int'l,* 521 F.3d at 1337. The '812 specification provides detail sufficient to perform this function, and is therefore entirely different from the rejected language in *Aristocrat/Int'l Game* because it provides a person of ordinary skill the necessary information to perform the recited function.

(b) "computer readable program code means for causing a computer to effect generating an IP datagram at a first partition of said multiple partitions to be forwarded to a second partition of said multiple partitions using a destination IP address" (claim 11)

PSI complains that IBM's corresponding structure "is described in even more vague and general terms." (PSI Brief 138.) Nothing could be further from the truth. For example, PSI refers to step 200 of Fig. 6 and col. 8:33-37 in isolation, and asserts that "no algorithm for 'construct[ing] the IP datagram,' as the disputed passages require, is provided in the above portions of the specification." (PSI Brief 138-139.) But PSI simply ignores IBM's other corresponding structure, namely Figs. 8A, 8B, col. 9:46-57 and col. 10:1-11. These portions of the specification describe how to construct an IP datagram.[22]

(c) "computer readable program code means for causing a computer to effect determining whether said destination IP address for said IP datagram comprises an IP address saved at said host-network interface for said at least one partition, and if so, forwarding the IP datagram directly from said first

[22] A person of ordinary skill at the time of the invention, having read the '812 specification would understand how the IP header, TCP header, and data are structured, as well as what is contained in the IP header or the TCP header. *See Aristocrat/Multimedia Games,* 2008 WL 484449, at *9 ("The evidence shows that modulation is a simple process that is well known in the art, and not some new development on the cutting edge of electrical and computer engineering. To ask a patentee to outline the steps for providing modulation would be to ask for a level of detail that is both unnecessary and problematic....").

partition to said second partition of said multiple partitions without employing said network" (claim 11)

PSI ignores certain portions of IBM's corresponding structure for this element. PSI refers to col. 8:20-30 in isolation, and asserts that this passage provides insufficient detail for one of ordinary skill in the art to practice the recited function. (PSI Brief 139-140.) However, PSI once again ignores IBM's other corresponding structure at col. 8:37-48 and Fig. 6, which describe in detail an embodiment of partition to partition communications without using a network. (*See* IBM Brief 76-77.)

PSI's misguided attempt to invalidate claim 11 should be rejected. *See AllVoice Computing PLC,* 504 F.3d at 1245 (defendant has the burden of establishing by "clear and convincing evidence" that "the disclosure[s] ... do not constitute sufficient structure to define the claim terms for the ordinarily skilled artisan").

B. The '002 "Firmware Booting" Patent

1. "firmware image" (claims 1, 9, 17)

PSI offers a construction for the term "firmware," alleges that IBM refuses to propose a construction for that term, and argues that "there is a 'live' dispute over the scope of the term 'firmware' that the Court should resolve." (PSI Brief 141.) But the term "firmware" never appears by itself in the '002 claims and is limited to use as a modifier for "firmware image" and "firmware image identifier."[23]

IBM proposes a construction for the term "firmware image." A construction of "firmware" by itself and not in the context of a "firmware image" is irrelevant and unnecessary.

IBM does not dispute PSI's proposed "firmware" construction so long as it is limited to its proper context. IBM further agrees that the inventors have acted as their own lexicographers for this term, but not in the way that PSI advocates. Instead, and as set forth in IBM's Opening Brief, the inventors specifically defined "partition management firmware," or what is also known as a "hypervisor." As illustrated below, the "definition" of "firmware" that PSI partially recites

from the '002 specification is taken out of context. The complete passage shows that this definition is specifically written for "partition management firmware" or a "hypervisor":

> Partition management firmware (hypervisor) 210 performs a number of functions and services for partitions 201-204 to create and enforce the partitioning of logically partitioned platform 200. Hypervisor 210 is a firmware implemented virtual machine identical to the underlying hardware. Firmware is "software" stored in a memory chip that holds its content without electrical power, such as, for example, read-only memory (ROM), programmable ROM (PROM), erasable programmable ROM (EPROM), electrically erasable programmable ROM (EEPROM), and non-volatile random access memory (non-volatile RAM). Thus, hypervisor 210 allows the simultaneous execution of independent OS images 201a-204a by virtualizing all the hardware resources of logically partitioned platform 200. Hypervisor 210 may attach I/O devices through I/O adapters 248-262 to single virtual machines in an exclusive mode for use by one of OS images 201a-204a.

('002 patent, col. 6:17-33.) Fig. 3 of the '002 patent illustrates this "partition management firmware":

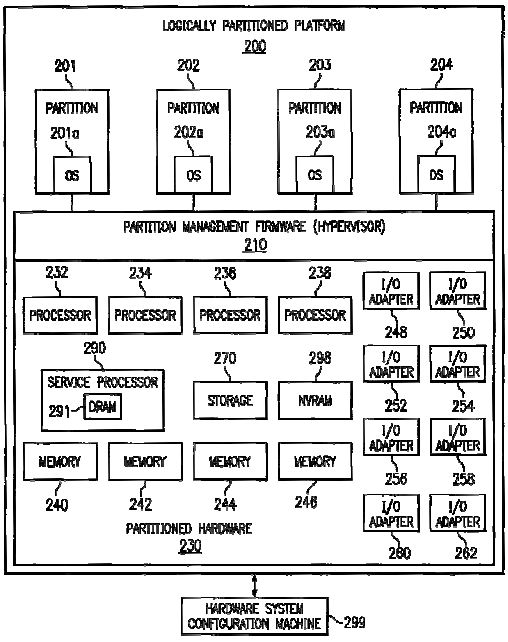

FIG. 3

The '002 specification separately describes the "firmware images," and illustrates on Fig. 3 that are distinct from the partition management firmware:

[23] The parties are not disputing the construction of the term "firmware image identifier."

> Partitioned hardware 230 also includes service processor 290. A non-volatile memory device 291, such as a DRAM device, is included within service processor 291. The partition tables and firmware images described herein, as well as other information, are stored within service processor memory 291.

('002 patent, col. 6:11-16.) Thus, the '002 inventors have indeed acted as their own lexicographers for the term "partition management firmware" or "hypervisor," but never did so for the terms "firmware" or "firmware image."

PSI completely ignores the intrinsic evidence regarding a "firmware image" and how it is used. (*See* '002 patent, col. 2:50-51; 2:61-63; 3:4-9; 6:12-16; 7:11-13; 7:40-41; *see also* IBM Brief 78-79.) That intrinsic evidence clearly shows that a firmware image is software copied from non-volatile memory and used to boot a partition, and this non-volatile memory is not limited to memory chips. PSI's sole reliance on its "firmware" definition (taken out of context as described above) to support this construction is inconsistent with the intrinsic evidence for "firmware image" and should be rejected.

PSI argues that "IBM's construction would destroy the very concept of firmware ...[because] [b]y expanding the concept of firmware to include software stored on any non-volatile media, including non-volatile media (like a floppy disk) that can be overwritten by any program or application, IBM's proposal runs contrary to the purpose of firmware." (PSI Brief 144.) But IBM's construction is for a "firmware image," not "firmware," and PSI has already conceded that "[a]n image is simply an 'instance' or 'copy' of the original." (PSI Brief 143.) A firmware image need not be stored in the way that PSI contends firmware itself should be stored, and the construction of "firmware image" should not be so limited.

PSI's extrinsic evidence for both "firmware" and "firmware image" does not support PSI's chip-only construction, and should be rejected.

Finally, PSI agrees that the firmware image is used to boot a partition, which is its only relevant use in the '002 claims (any firmware uses are irrelevant because none are claimed). This undisputed function of booting the partition is consistent with the '002 claims and provides context to the jury.

Thus, IBM's construction should be adopted.

2. "capable of being executed during a power-on process to boot said computer system" (claims 1, 9, 17)

The parties' only dispute is whether the power-on process includes executing the partition firmware.

As set forth in IBM's Opening Brief, the plain language of claim 1 ("each one of said plurality of different firmware images capable of being executed during a power-on process") illustrates that executing a firmware image must be part of the power-on process. PSI completely ignores this intrinsic evidence, presumably because it directly contradicts its construction. This alone should end the Court's inquiry.

Both parties agree that the power-on process is illustrated by steps 402-430 of the Fig. 4 flowchart. (*See* PSI Brief 146-147.) However, PSI contends that the power-on process does not include the last step of Fig. 4, step 430, which consists of executing the partition firmware. In other words, PSI is arguing that the power-on process includes steps 402-428 but not step 430, even though it is part of the power-on process. This is completely illogical.

PSI also argues that the '002 claims require re-booting a partition (*i.e.*, executing partition firmware) before booting an operating system. To simplify the issue and hopefully resolve the dispute, IBM is willing to modify its construction to the following: "capable of being executed during the operations performed by a computer system from the time it is turned on until and including executing the partition firmware to boot said computer system." This is consistent with the '002 claims, which recite that re-booting a partition occurs before booting the operating system.

Accordingly, IBM's construction should be adopted.

3. "instruction means for storing a plurality of different firmware images in said computer system" (claim 9)

In its Opening Brief, IBM explained the difference in the parties' constructions for the function, and indicated why IBM's construction is more appropriate. (IBM Brief 83.) PSI makes no attempt to refute IBM's construction.

PSI misrepresents IBM's recitation of corresponding structure for this claim. IBM is not proposing "***all*** programming that causes a general purpose computer to perform the function recited in the claim." (PSI Brief 149). Instead, IBM is proposing a computer program product, the structure of which is described at col. 8:4-20 and equivalents thereof, followed by the recited function and, in parentheses, the algorithm for performing the recited function through software on that computer program product.

The '002 specification describes the structure of the computer program product at col. 8:4-20. It is irrelevant whether this language appears in other IBM patents because it recites structure for the computer program product in this patent. A person of ordinary skill in the art at the time of the invention would certainly understand the structure for a "computer readable media" such as RAM, CD-ROMs, and DVD-ROMs.

PSI's statement that col. 8:4-20 "says nothing at all about the particular algorithm needed to accomplish the claimed function" is misleading. (PSI Brief 149.) IBM's proposed corresponding structure includes this recitation of structure, ***along*** with the algorithm for the software on that structure that performs the recited function. PSI ignores the second part of IBM's construction when addressing the first part, and vice-versa.

PSI claims that IBM's disclosure of the algorithm for performing the recited function (*e.g.*, Fig. 3 and col. 6:13-16) is inadequate. The '002 specification explains that one embodiment for storing different firmware images is to store them at service processor memory 291, illustrated in Fig. 3. PSI complains that this does nothing more than describe the claimed function, but a person of ordinary skill at the time of the invention would have understood how to save a firmware image to a designated processor's memory without any further explanation. *Aristocrat/Multimedia Games,* 2008 WL 484449 at *9. The detail provided by the '002 specification for performing this function goes beyond the rejected language in *Aristocrat/Int'l Game*, and provides a person of ordinary skill in the art the necessary information to perform the recited function.

4. "instruction means for rebooting one of said plurality of partitions utilizing one of said plurality of firmware images without rebooting other ones of said plurality of partitions" (claim 9)

IBM is not proposing "any computer program" as corresponding structure, but rather the software stored in a computer readable media and defined by the algorithm set forth in the '002 specification.

PSI argues that IBM's disclosure of the algorithm from the '002 specification fails to meet the standard set forth in *Aristocrat/Int'l Game*. PSI then quotes some of IBM's cited disclosures and highlights purported inadequacies, such as how a partition table is checked. (PSI Brief 151-153.) PSI concludes by stating that "[f]ar from a specific algorithm, as *Aristocrat/Int'l Game* requires, the specification gives only the general outlines of the phases in the rebooting process." *Id.* at 153.

The algorithm provided in the '002 specification is much more detailed than the patentee's disclosure in *Aristocrat/Int'l Game*. The '002 specification sets forth a detailed flowchart (Fig. 4) for performing the recited function, along with an explanation of how to perform each step (col. 7:30-45). PSI's complaints ignore the Federal Circuit's caveat to the algorithm requirement in *Aristocrat/Int'l Game*:

> It is certainly true that the sufficiency of the disclosure of algorithmic structure must be judged in light of what one of ordinary skill in the art would understand the disclosure to impart.

Aristocrat/Int'l Game, 521 F.3d at 1337. The purported inadequacies raised by PSI, such as how a partition table is checked, would be understood by a person or ordinary skill in the art.

The Court should reject PSI's attempt to invalidate this claim. *See AllVoice Computing PLC,* 504 F.3d at 1245.

VII. THE '851 I/O PATENT

1. "input/output control blocks" (claims 9, 21, 22)

PSI's construction adds unnecessary structural limitations by attempting to: (1) limit control blocks to a "hardware or microprogramming construct;" and (2) limit the type of information contained in the control blocks. PSI selectively quotes from the '851 specification to

support its proposed limitations. And while PSI is quick to refer to "black letter law," it improperly reads in limitations from the specification – in direct contravention of the Federal Circuit law. *See Phillips*, 415 F.3d at 1323.

Input/output control blocks are data structures in memory. They are not, however, limited to hardware or microprogramming constructs. For example, the '851 specification states:

> Each of the different channel images is represented by a channel control block (CHCB) in the I/O subsystem storage. The I/O subsystem storage is preferably in a memory area separate from system main storage, so that the I/O subsystem storage is not addressable by CPU instructions, but is addressable by internally coded (microcoded) instructions, and hardware.

('851 patent, col. 14:62-68; *see also*, col. 12:16-21; col. 21:16 -32.) A person of ordinary skill would understand this to mean that the control block is simply a type of data structure resident in the system memory of an IBM S/390.

Moreover, a data structure is a way of storing data in a computer. From the point of view of the data, it is irrelevant whether that data is manipulated under the control of hardware, software, or some combination of both. The claimed invention of the '851 patent provides a data structure that is used to share I/O channels and I/O devices, and whether that data structure is managed by hardware, software, or some combination of both is tangential to the nature of the invention; it is merely an implementation detail. As it has from the beginning to the end of its brief, PSI is seeking to excuse it copying of the broader IBM invention by attempting to artificially restrict the claim language to a particular implementation.

PSI also limits its construction of input/output control blocks to specify "a shared resource to an OS, and may be said to represent an image of the resource to each sharing OS." (PSI Brief 155.) This limitation, incorporated from the '851 specification, contradicts other parts of the specification that PSI chooses not to address. For example, the '851 patent states that:

> In this invention, ***non-sharing resource control blocks*** (like those found in prior systems) may also be intermixed with sharable resources of the same type. ***Thus, a non-shared subchannel (SCB) may be used for an I/O device*** dedicated to a single OS, and sharable subchannels (SSCBs) may also be used for enabling passthru I/O operations by plural OSs to that device.

(*Id.*, col. 8:62-68, emphasis added.) PSI's limitation of input/output control blocks to specify only shared resources is inconsistent with the intrinsic evidence.

Furthermore, PSI attempts to limit input/output control blocks so that they only specify resources to an operating system ("OS"). The '851 specification, however, states that "Each control block in a sharing set is assigned to a different OS by means of a novel 'image identifier' (IID). ***The hypervisor may also be assigned a control block*** in a sharing set." (*Id.*, col. 8:3-7, emphasis added.) A person of ordinary skill in the art at the time of the invention would have understood that a hypervisor is distinct from an OS.

PSI goes a step further, arguing that IBM's construction would encompass "a PDF of a user's manual for a printer that one might store on a hard drive, since the PDF is a 'data structure' and the printer user manual is 'information about I/O resources.'" (PSI Brief 157.) However, PSI provides no citation to the specification and claim language to support this absurd argument. One of ordinary skill in the art would understand that I/O control blocks, as construed by IBM ("data structures containing information about I/O resources"), are not read in a vacuum. PSI's example, and overall argument on this point, fails because a person of ordinary skill in the art would not read claim 1 of the '851 patent and possibly think that it covers a user printer manual or a PDF. Nor would that person understand that PDFs have input/output resource identifiers or image identifiers, or that the PDFs would be accessed using their image identifiers.

PSI's misapplies *Honeywell Int'l, Inc. v. ITT Industries*, 452 F.3d 1312 (Fed. Cir. 2006) to read limitations from the specification into the claims. In *Honeywell*, the Court narrowed the scope of the claims where the patentee drafted the claims to cover devices not mentioned in the specification. *Id.* at 1318. Furthermore, the patent claims did not include any part of the prior art problem that the invention was designed to address. *Id*. In the patent there, "the 'invention' was identified to be only a fuel filter." *Id*. at 1316. The patentee repeatedly referred to the "fuel filter as 'this invention' or 'the present invention,'" and then attempted to claim broader protection for a fuel injection system component that would include a fuel filter. *Id*. at 1318. In contrast, the '851 specification never characterizes the invention as either control blocks or "a hardware or

microprogramming construct," as PSI contends. Rather, the '851 patent states that "[t]he invention provides a novel method for sharing I/O channels" ('851 patent, col. 7:43-44), and "[a] method for sharing input/output (I/O) resources among a plurality of programs in a computer electronic complex." *Id*., col. 32:4-6.[24] Control blocks are data structures ***used*** by the invention, not the invention itself. In one embodiment they may be "hardware or micro-programming constructs." But the '851 patent makes clear that "it should be understood that the above described embodiments have been provided by way of example rather than as a limitation." *Id*., col. 31:68-32:2.

PSI's reliance on *SciMed Life Systems, Inc. v. Advanced Cardiovascular Systems, Inc.*, 242 F.3d 1337 (Fed. Cir. 2001) is equally misplaced. In *SciMed* the court relied on language in the specification to conclude that the claims covered only one (coaxial/sleeve) of two possible ways for making a catheter (the other being a lumen catheter). *Id.* PSI neglects to mention that in *SciMed* the patent at issue stated "[t]he intermediate sleeve structure defined above is the basic sleeve structure for ***all embodiments of the present invention contemplated and disclosed herein***." *Id*. at 1343 (emphasis in original). The court held that:

> The words 'all embodiments of the present invention' are broad and unequivocal. It is difficult to imagine how the patents could have been clearer in making the point that the coaxial lumen configuration was a necessary element of every variant of the claimed invention.... This is a clear case of disclaimer of subject matter, ***that absent the disclaimer, could have been considered to fall within the scope of the claim language***.

Id. at 1344 (emphasis added). PSI cites *SciMed* for the proposition that "the characterization of the coaxial configuration as part of the 'present invention' is strong evidence that the claims should not be read to encompass the opposite structure." (PSI Brief 156.) However *SciMed* limited the claims due to a strong disclaimer of subject matter. *SciMed*, 242 F.3d at 1344. Here PSI fails to mention the glaring absence of any such disclaimer in the '851 patent for input/output

[24] Contrast this with the claim discussed in *Honeywell*, "Moldable material for fuel system components...," *Honeywell*, 452 F.3d at 1315 n. 1, where the disputed characterization of the invention, the fuel system components, was used in the claim language to describe the invention.

control blocks. Furthermore, the '851 patent explicitly states that "it should be understood that the above described embodiments have been provided as way of example rather than as a limitation." (‘851 patent, col. 31:68-32:2.)

Finally, a person of ordinary skill in the art would have understood that there are many embodiments of the input/output control blocks, such as hardware or microprogramming constructs. Unlike *SciMed*, where there were only two possible embodiments for catheters and the patentee chose one, there are several possible embodiments for the control block data structures in the '851 patent. There is no black or white choice as there was in *SciMed*.

Accordingly, IBM's construction for "input/output control blocks” is more appropriate and should be adopted.

VIII. CONCLUSION

For the reasons stated above and in IBM's Opening Brief, IBM respectfully requests that the Court adopt its proposed claim constructions.

Dated: June 27, 2008

QUINN EMANUEL URQUHART OLIVER & HEDGES, LLP

By: /s/ Richard I. Werder, Jr.

Richard I. Werder, Jr.
Edward J. DeFranco
Richard W. Erwine
Alexander Rudis
51 Madison Avenue
22nd Floor
New York, New York 10010-1601
(212) 849-7000

Frederick A. Lorig
865 S. Figueroa Street
Los Angeles, California 90017
(213) 443-3000

Attorneys for Plaintiff
International Business Machines Corporation